

AD-A107 254

AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH
ANALYSIS OF FLOW BEHAVIOR WITHIN AN INTEGRATED COMPUTER-COMMUNI--ETC(U)
MAY 79 C A CLABAUGH
AFIT-C1-79-2130

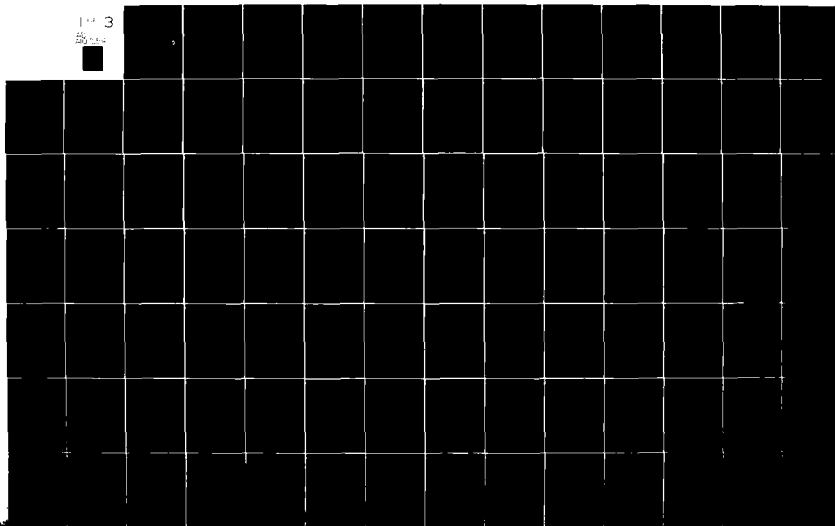
F/G 17/2

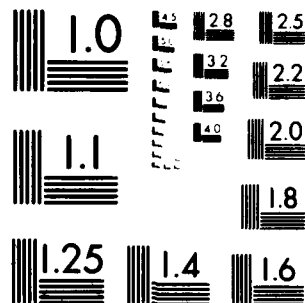
UNCLASSIFIED

NL

1 of 3

5/79





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 79-213D	2. GOVT ACCESSION NO. AD-A107 254	3. RECIPIENT'S CATALOG NUMBER 254
4. TITLE (and Subtitle) Analysis of Flow Behavior Within An Integrated Computer-Communication Network.		5. TYPE OF REPORT & PERIOD COVERED THESIS/DISSERTATION
7. AUTHOR(s) Major Carroll Ardee/Clabaugh		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: Texas A&M University		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 15077
14. MONITORING AGENCY NAME & ADDRESS (If different from Controlling Office) LEVEL		12. REPORT DATE 10 May 1979
		13. NUMBER OF PAGES 255
		15. SECURITY CLASS. (of this report) UNCLASS
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED 174727-1-77-110		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES APPROVED FOR PUBLIC RELEASE: IAW AFR 190-17 FREDRIC C. LYNCH, Major, USAF Director of Public Affairs Air Force Institute of Technology (ATC) Wright-Patterson AFB, OH 45433		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) ATTACHED		

DTIC
ELECTE
NOV 6 1981

22 OCT 1981

Fredric C. Lynch
FREDRIC C. LYNCH, Major, USAF
Director of Public Affairs
Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DTIC FILE COPY

AD A 707 254

ABSTRACT

Analysis of Flow Behavior Within an Integrated
Computer-Communication Network. (May 1979)

Carroll Ardee Clabaugh, B.A., Coe College;

M.S., New Mexico State University

Chairman of Advisory Committee: Dr. Udo Pooch

This research concerns itself with the flow behavior in a computer-communication network that integrates data and voice classes of traffic. This behavior is analyzed as a queueing problem by modeling nodes and channels and evaluating their behavior under different arrival rates.

Switching mechanisms, flow control considerations, and the development and use of simulation and analytical models are emphasized.

Prominent flow control strategies are classified into a framework for potential utility as regulating controls in an integrated environment.

An event-driven FORTRAN network simulator is developed based on common call management, traffic integration on trunk lines, and an underlying circuit switched communications subnet. The network simulator is a cost-effective portable tool that has application in an environment where computer resources are limited. It is sufficiently general in structure so that several design factors can be varied by the user to evaluate network performance by collecting

appropriate empirical data.

An algorithm is formulated that permits decomposition of the network into nodal configurations. A FORTRAN program is developed that implements this algorithm..

Finally, summary information demonstrates the practicality of an integrated network and need for flow controls.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Avail and/or	
Dist Special	
A	

79-213D

ANALYSIS OF FLOW BEHAVIOR WITHIN
AN INTEGRATED COMPUTER-COMMUNICATION NETWORK

A Dissertation
by
CARROLL ARDEE CLABAUGH

Approved as to style and content by:

Udo W. Poth
(Chairman of Committee)

Newton C. Eller
(Head of Department)

Dan Dren
(Member)

SK Jones
(Member)

V. L. Rhine
(Member)

May 1979

ANALYSIS OF FLOW BEHAVIOR WITHIN
AN INTEGRATED COMPUTER-COMMUNICATION NETWORK

A Dissertation
by
CARROLL ARDEE CLABAUGH

Submitted to the Graduate College of
Texas A&M University
in partial fulfillment of the requirement for the degree of
DOCTOR OF PHILOSOPHY

May 1979

Major Subject: Computing Science

811030054

ANALYSIS OF FLOW BEHAVIOR WITHIN
AN INTEGRATED COMPUTER-COMMUNICATION NETWORK

A Dissertation
by
CARROLL ARDEE CLABAUGH

Approved as to style and content by:

Udo W. Poth
(Chairman of Committee)

Newton C. Eller
(Head of Department)

Dan Drew
(Member)

St. Jones
(Member)

V. L. Rhine
(Member)

May 1979

ABSTRACT

Analysis of Flow Behavior Within an Integrated
Computer-Communication Network. (May 1979)

Carroll Ardee Clabaugh, B.A., Coe College;

M.S., New Mexico State University

Chairman of Advisory Committee: Dr. Udo Pooch

This research concerns itself with the flow behavior in a computer-communication network that integrates data and voice classes of traffic. This behavior is analyzed as a queueing problem by modeling nodes and channels and evaluating their behavior under different arrival rates.

Switching mechanisms, flow control considerations, and the development and use of simulation and analytical models are emphasized.

Prominent flow control strategies are classified into a framework for potential utility as regulating controls in an integrated environment.

An event-driven FORTRAN network simulator is developed based on common call management, traffic integration on trunk lines, and an underlying circuit switched communications subnet. The network simulator is a cost-effective portable tool that has application in an environment where computer resources are limited. It is sufficiently general in structure so that several design factors can be varied by the user to evaluate network performance by collecting

appropriate empirical data.

An algorithm is formulated that permits decomposition of the network into nodal configurations. A FORTRAN program is developed that implements this algorithm.

Finally, summary information demonstrates the practicality of an integrated network and need for flow controls.

ACKNOWLEDGEMENTS

I am most appreciative of the support, direction, and constructive criticism provided by my chairman, Dr. Udo Poch, throughout my research effort. His professional assistance has contributed greatly to timely completion of this dissertation. Also, much gratitude is due Drs. Richard Feldman and Stephen Jones for their assistance in the development of a queueing model for an integrated communications node.

Particular thanks are due my wife, Joyce, who typed my dissertation and assisted me throughout my effort.

To my wife, Joyce, and children
Christopher, Lonnie, Sherie, and Erin
for their patience and understanding.

TABLE OF CONTENTS

	Page
ABSTRACT.....	iii
ACKNOWLEDGEMENTS.....	v
DEDICATION.....	vi
TABLE OF CONTENTS.....	vii
LIST OF FIGURES.....	xi
LIST OF TABLES.....	xii
Chapter	
I. INTRODUCTION.....	1
1. Computer-Communications Networking.....	1
1.1 Evolutionary Development.....	2
1.2 The Case for an Integrated Packet/Circuit Switch Network.....	3
1.3 Research Objectives and Plans.....	4
II. LITERATURE SURVEY.....	8
2. Overview.....	8
2.1 Network Design Areas.....	8
2.1.1 Integration Methodology.....	9
2.1.2 Line Switching.....	12
2.1.3 Routing.....	15
2.1.4 Flow Control.....	16
2.1.5 Nodal Design.....	18
2.1.5.1 Circuit Switch Architecture..	18
2.1.5.2 Packet Switch Architecture...	22
2.1.6 Priority/Preemption Controls.....	24
III. FLOW CONTROLS IN DISTRIBUTED COMPUTER- COMMUNICATION NETWORKS.....	26
3. Problem Definition.....	26
3.1 Structural Flow Control Layers.....	27
3.2 Overview of a Flow Control Strategy.....	29
3.3 Objectives of a Flow Control Strategy.....	30
3.3.1 Congestion Prevention.....	30
3.3.2 Deadlock Prevention.....	30
3.3.3 End-to-End Flow Control.....	30

	Page
3.3.4 Fair Allocation of Communication Resources.....	31
3.4 Flow Control Strategies.....	31
3.4.1 Subnet Controlling.....	31
3.4.1.1 Isarithmic.....	33
3.4.1.2 Centralized Flow Control Station.....	34
3.4.1.3 Flow Control Tables.....	35
3.4.1.4 Nodal Buffer Monopoly.....	36
3.4.1.5 Overflow Detection.....	38
3.4.1.6 Nodal Limits.....	39
3.4.1.7 Node-Node ARQ (Automatic Request for Repeat).....	39
3.4.1.8 Dedicated Nodal Buffers.....	40
3.4.2 Subscriber Controlling.....	40
3.4.2.1 Reservation Schemes.....	41
3.4.2.2 Crate Occupancy.....	43
3.4.2.3 Subscriber-Subscriber.....	44
3.5 Conclusions.....	46
IV. THE NETWORK SIMULATION MODEL.....	48
4. Introduction.....	48
4.1 Description of the Queueing Model.....	54
4.2 Overview of the Simulator.....	56
4.3 Properties of the Model.....	60
4.3.1 Operating Characteristics.....	60
4.3.2 Description of Functional Modules.....	60
4.3.2.1 Initialization.....	61
4.3.2.2 Transaction Generation.....	61
4.3.2.3 Arrival.....	61
4.3.2.4 Routing.....	62
4.3.2.5 Departure.....	63
4.3.2.6 Event Selection.....	63
4.3.2.7 Statistical Gathering and Output.....	64
4.4 Description of Major Tables.....	64
V. ANALYTIC DEVELOPMENT OF AN INTEGRATED MODEL.....	66
5. Introduction.....	66
5.1 Solution Process.....	69
5.1.1 Algorithm Development.....	73
5.1.2 P_0 Vector Determination.....	76
5.1.3 Steady-State P Vector Determination.....	76

	Page
5.2 Determination of Effectiveness Measures.....	82
5.3 Conclusions.....	83
VI. DEVELOPMENT OF AN INTEGRATED FLOW CONTROL SCHEME..	85
6. The Need For a Flow Control Scheme.....	85
6.1 Utility of Existing Strategies.....	85
6.2 A Ring Control Scheme Proposal.....	86
6.2.1 Nodal Limits Ring.....	88
6.2.2 Nodal Monopoly Ring.....	88
6.2.3 Window Ring.....	89
6.3 Ring Scheme Implementation.....	89
6.3.1 Nodal Limit Variables.....	90
6.3.2 Global Monopoly Matrix.....	90
6.4 Subscriber Window Development.....	90
6.4.1 Multiple Interrupt Level Capability...	92
6.4.2 Variable I/O Data Rate Terminations...	92
6.5 Subscriber Software Development.....	92
VII. ANALYSIS OF NETWORK BEHAVIOR.....	93
7. Network Configuration.....	93
7.1 Examination of the Simulator.....	93
7.1.2 Confidence in the Simulator.....	95
7.1.3 Description of Simulator Output.....	100
7.2 Analysis of Network Flow.....	100
7.3 Analysis of Analytic Output.....	101
7.3.1 Decomposition-Synthesis Approach.....	108
7.3.2 Utility of the Analytic Model.....	108
7.4 Summary.....	109
VIII. CONCLUSIONS AND RECOMMENDATIONS.....	111
8. Conclusions.....	111
8.1 Recommendations.....	112
REFERENCES.....	114
APPENDIX A.....	122
APPENDIX B.....	177
APPENDIX C.....	203
APPENDIX D.....	216

x

	Page
APPENDIX E.....	240
VITA.....	255

LIST OF FIGURES

Figure	Page
1. Allocation schemes.....	11
2. PCM switching.....	14
3. Circuit switch T1 terminations.....	20
4. Buffer state matrix.....	37
5. Network configuration.....	49
6. Master frame clocking.....	51
7. Network nodal model.....	57
8. Rate matrix for three channel system.....	68
9. Rate matrix block diagram.....	70
10. Flow control ring scheme.....	87
11. Flow control monopoly matrix.....	91
12. 10-Node network configuration.....	94
13. 10-Node routing table initialization.....	96
14. Input parameters and specifications for 10-node network.....	97
15. Packet delay.....	102
16. Link utilization.....	103
17. Packet throughput.....	104
18. Class I blocking.....	105
19. Calls in system.....	106
20. Data transactions in system.....	107

LIST OF TABLES

Table	Page
I. Classification of Flow Control Strategies.....	32
II. Calculation of Confidence Intervals.....	99
A2-1. Parameter (PARAM) [X].....	126
A2-2. Event (EVTBL) [Node, Entry].....	127
A2-3. Destination (DESTAB) [Node, Dest].....	128
A2-4. Channel Table (CHANTB) [Channel, Entry].....	129
A2-5. Queue (QUEUE) [Node, Entry].....	130
A2-6. Call Queue (CALLQ) [Knode, Entry].....	131
A2-7. Cumulative Time (CUMTIM) [Node, Entry].....	132
A2-8. Calls Accepted/Rejected (CALLS) [Knode, Entry]...	133
A2-9. Link (LINKTB) [Node, Dest].....	134
A2-10. Seed (SEEDTB) [Node, Distribution].....	135
A2-11. Link Availability (NLINES) [Channel].....	136
A2-12. Queue Entry Count (QCNT) [Node].....	137
A2-13. Channel Connectivity (NODCHL) [Link].....	138
A2-14. Alternate Channel (ALTCH) [Channel].....	139
A2-15. Circuit Switch Arrivals (CSARV) [Knode, Entry]...	140
A2-16. Alternate Destination (DSTALT) [Node, Dest].....	141
E1. Arrival Pattern 1.....	241
E2. Arrival Pattern 2.....	242
E3. Arrival Pattern 3.....	243
E4. Arrival Pattern 4.....	244
E5. Arrival Pattern 5.....	245
E6. Arrival Pattern 6.....	246

Table	Page
E7. Arrival Pattern 7.....	247
E8. Arrival Pattern 8.....	248
E9. Arrival Pattern 9.....	249
E10. Arrival Pattern 10.....	250
E11. Arrival Pattern 11.....	251
E12. Arrival Pattern 12.....	252
E13. Arrival Pattern 13.....	253
E14. Arrival Pattern 14.....	254

CHAPTER I

INTRODUCTION

1. Computer-Communications Networking

The steady growth of the data processing industry and the telephone network over the last four decades have jointly introduced the remote sharing of information data bases as a cost-effective management tool. The power of computer facilities has been made available to classes of users through terminals connected to these facilities over a communications subnet comprised of communication processors (nodes) and common user circuitry (links). Computer-communication networks are created by this marriage to satisfy general networking requirements for real time transaction power, sharing of information and resources such as terminals, applications, and facilities, distributed data base processing, and load balancing. Branscomb [4] suggested the following three significant reasons for the overall success and cost-effectiveness of these networks:

- (1) Increasing performance for decreasing costs of memory, storage, processors, and communications.
- (2) Increasing capability and reliability of computer programs.
- (3) Increasing people costs.

The Communications of the Association for Computing Machinery is used as a pattern for format and style.

In a computer-communications network, the proper allocation of subnet resources such as communication links, switching nodes, processing units, and network storage elements insures uniform facility utilization, stable system operation, and acceptable user service. Network sharing resources are defined as heterogeneous computer systems called hosts that are geographically distributed and interconnected by the communications subnet. Within a computer-communications network are several layers of protocol (rules) necessary to escort each information exchange from its originating node to the destination node. Existing networks are designed in a variety of configurations, which include centralized, distributed, ring, or various combinations of these [16, 74].

1.1 Evolutionary Development

The Department of Defense Advanced Research Projects Agency (ARPA Network (ARPANET)) is the predecessor of existing computer-communications networks. It was conceived with the concept of fast and reliable transportation of data packets (1000 bits per packet) from a network source to a destination through the communications subnet. These packets are independently routed from node to node on a store and forward basis until the final subnet node is reached. They are then rearranged and delivered to the "host" system. The ARPANET still exists. It links together several university and research agencies. It has become an operational network for continued research and design into

the myriad of problems associated with the implementation of computer-communications networks.

Since the inception of ARPANET in the early 1960's, numerous other national and international networks have been designed and implemented [74, 89]. This proliferation and use of ARPANET technology has proven computer networking to be cost-effective, and has become a valuable management tool for satisfying needs of the information processing industry. The importance of computer networking, and its impact on both private and public sectors is widely recognized [74].

1.2 The Case for an Integrated Packet/Circuit Switch Network

Recent Defense Communication Agency (DCA) studies have shown the desirability of an all-digital, switched network which integrates voice, interactive and bulk data for the 1980's [80, 90]. Several additional studies relating to information processing growth in the next few decades portend new services with substantially increased data flows [4, 76, 79, 83]. As stated by Ross [83], "the spectrum of terminals requiring service is expected to range from 45 bps TTY terminals and 2400 bps vocoders to interactive graphics, digital facsimile, and slow-scan video terminals requiring rates in the tens and hundreds of kilobits per second". To facilitate the implementation of these classes of traffic, today's and future computer-communications networks use or plan to use Circuit Switching (CS) or Packet Switching (PS)

techniques for the movement of data through the subnet. In a circuit switched network a physical end-to-end path, as opposed to the store-and-forwarding process of packet networks, is found before communication circuits and buffers are committed for the duration of the connection. Both network types offer a wide range of advantages and disadvantages depending on factors such as class of data traffic, user applications, interface complexity, and user transparency. The projected diversity in terminal devices, growth of new services, and advances in satellite communications, Large Scale Integration (LSI) technology, digital switching and transmission capabilities, give rise to the potential sharing of transmission capacity, switching equipment, and common network control and signalling to reduce network costs. The need for a uniform network is voiced by management and system designers concerned with limitations of existing networks for projected information processing requirements and services [3, 8, 10, 19, 21, 25, 41, 51, 54, 60, 69, 79, 84, 88].

1.3 Research Objectives and Plans

The research of this dissertation investigates the flow of data and voice in an integrated network. The research is based on the premise that unlike existing store-and-forward distributed networks, an integrated network based on circuit switching techniques can significantly reduce the subnet

overhead and control instabilities inherent in such networks, facilitate increased interoperability between existing and projected new services, and place the burden of flow control solely at end-point nodes where more efficient controls can be exerted. The development of performance evaluation and control tools are primary research objectives. Switching mechanisms, flow control considerations, and the design of analytical and simulation models are emphasized.

A literature review of architectural considerations and developments is provided in Chapter II. Detailed concepts and terminology are presented as support material to succeeding chapters.

Prior research efforts have investigated architectural nodal designs, cost trade-offs between circuit and packet network types, and the feasibility of a uniform network. There lacks (1) an efficient analytical tool or simulation model that characterizes a network based on circuit switching techniques and (2) flow controls that are applicable to such a network. This research systematically investigates and classifies existing circuit/packet flow control strategies into a framework for potential network integration. Chapter III presents this classification.

The lack of an existing model has required extensive design to obtain a usable network simulator. A discrete event-driven FORTRAN simulator is implemented that integrates common data/voice call management, traffic on trunk lines,

and an underlying circuit switched communications subnet.

Its developmental goals are two-fold:

- (1) To be sufficiently general in structure so that several design factors can be varied by the user to evaluate network performance by collecting empirical data, and
- (2) To be a cost-effective highly portable tool that has application in an environment where computer resources are limited.

Chapter IV details this development.

The need for analytic representation of the network leads to the formulation of an algorithm for analytic representation of nodal behavior. This algorithm permits the decomposition of the network to determine nodal steady-state measures of effectiveness. It is based on (1) an application of a rate matrix from queueing theory, and (2) similarity transformations involving eigenvalues and eigenvectors using linear matrix concepts. Chapter V presents the development of this algorithm. A ring control hierarchy for network nodal flow control is presented in Chapter VI. Analysis of network flow behavior is outlined in Chapter VII.

Finally, a summary of the results of the research, conclusions, and proposed recommendations and suggestions for further research are presented in Chapter VIII. Applicable simulation data and program listings of the

simulator and analytic model are contained in Appendices A through E.

CHAPTER II

LITERATURE SURVEY

2. Overview

Design implementations for communications processing in data networks have been based on analog transmission and either circuit or store-and-forward switching principles. The gradual emergence of highly efficient digital transmission paths, projected requirements for speech, data, and video [4, 54, 79], and the increased utilization of Time Division Multiplexing (TDM) techniques provide motivation for a uniform system design to handle all classes of traffic. Cicchetti and Lubarsky summarized several unique design problems which an integrated voice/data system must address [8]:

- (1) Common calling establishment/disestablishment procedures,
- (2) an order of magnitude increase in traffic volume at the nodes,
- (3) end-to-end network and error controls,
- (4) priority/preemption controls for voice and data, and
- (5) the digitization of all traffic.

2.1 Network Design Areas

The underlying structure of an integrated network will

rely on circuit, store-and-forward, or hybrid switching. There appears to be disagreement on which structure offers the most promise [8, 10, 18, 20, 25, 31, 41, 46, 47, 57, 58, 69, 79, 88, 90, 99, 100]. Independent of the underlying switching structure, the major problems unique to an integrated network architecture can be categorized into the following design areas:

- (a) Integration Methodology
- (b) Line Switching
- (c) Routing
- (d) Flow Control
- (e) Nodal Design
- (f) Priority/Preemption Controls

2.1.1 Integration Methodology

The two major forms for transmitting data in use today are analog and digital. Analog transmission techniques represent a voice or data transaction by an analog signal continuously varying in voltage or current with respect to time. The amplitude represents the information. Using digital transmission techniques, all data is represented as a discrete code with constant amplitude. The information is retained on a serial time basis. Digital voice is commonly obtained by using Pulse Code Modulation (PCM) to convert voice samples to a binary code of several bits [68].

Although analog transmission has been the mainstay of communications processing, the increasing requirements for

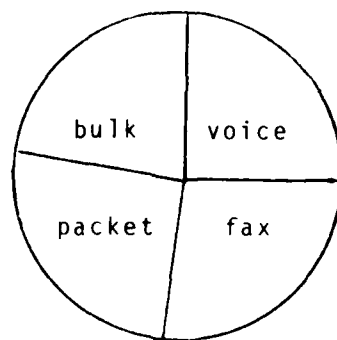
transmitting digital data and interfacing to digital equipment, coupled with the advantages of PCM switching, have resulted in a gradual shift to digital transmission implementation [97].

The family of Bell System T-Carrier Transmission Systems form the foundation for digital transmission. Two levels of T-Carrier Systems predominate today, the T1 for short distance transmission (50 miles) and the T2 for distances up to 500 miles. Each T1 carrier provides a link capacity of 1.544 megabits per second (which is subdivided into multiple Half Duplex (HDX) channels), and represents the major building block for any integrated methodology design.

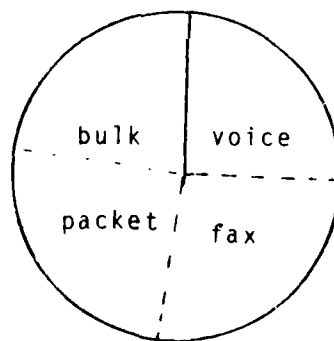
Integration methodology is concerned with the allocation of the T1 link into multiple subframes, each containing time slot packets of data, whether the data be facsimile, bulk, message, packet, etc., (Fig. 1) With a fixed allocation scheme, a specific number of time slots are reserved for each class, guaranteeing that each is given a prescribed grade-of-service (Fig. 1a). A major limitation of this design is the inability to make use of unused link capacity. A dynamic allocation scheme incorporates a movable frame boundary which permits unused slots in one class to be made available as slots to other classes of data (Fig. 1b). This scheme results in a better utilization of the link capacity, but requires considerable logic to implement [21, 31]. The research model will investigate a third allocation approach -

Fig. 1. Allocation schemes

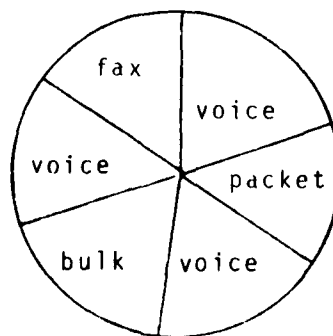
(a) Fixed boundary



(b) Movable boundary



(c) Competitive allocation



competitive allocation (Fig. 1c). Using this approach, classes of data compete for time slots using a first-come-first-served (FCFS) scheme. This will insure maximum usage of each frame; in addition, the need for various types of boundary controls will be investigated by statistically measuring subscriber (end user) delays and traffic flow.

2.1.2 Line Switching

Sanders defines digital line switching as the process whereby a digital switch receives data, stores it for a period of time, and forwards it to another switch over some channel [88]. Data network switching design is based on circuit or store-and-forward principles. Store-and-forward switches (nodes) group bits into packets or messages (multiple packets), and route them among each other using nodal tables and a network routing algorithm until the final destination is reached. When a switch receives a packet or a message, an acknowledgment is returned to the sending node and on-going switching responsibility is assumed. Since the path from source to destination is not a physical connection, queues may build at the various switching nodes, creating variable delays. There have been numerous data networks designed with store-and-forward switching, with corresponding performance evaluation widely documented [6, 9, 18, 34, 49, 50, 56, 67, 72, 81, 82, 86]. A circuit switching design involves the transmission of data through a nodal crosspoint structure to effect a link between two channels. The

replacement of analog electromechanical switching elements such as step by step, crossbar, relay, correes, etc., with digital Medium Scale Integration (MSI) SN7454, SN74150, and SN74200 chips as the crosspoint elements provides faster switching and reduced cross network delay [27, 28, 55, 68, 98]. Commercial digital circuit switches are now available combining TDM principles with PCM switching using MSI logic and Random Access Memory (RAM) as crosspoint elements to produce nanosecond switching speeds [39, 68]. The implementation of PCM switching requires an information memory to provide fixed channel assignments for both transmit and receive inlets, and a connection memory which contains the stored routing information for the connection (Fig. 2). Physical circuits (Full Duplex (FDX) channels) are thus established between subscribers and maintained for the duration of a transaction. As a result, no queues build up along the route. Performance comparisons between circuit and store-and-forward switching show that the crossover efficiency is highly dependent on the implementation structure [18, 58, 81, 82, 87, 88].

The switching design in a uniform system should be the most efficient structure for moving data among multiple T1 links and nodes. A research model is designed and implemented on the premise that an underlying digital circuit switch structure offers more potential, since microtechnology has made subscriber-to-subscriber control designs cost effective alternatives to subnet controls [88].

2.1.3 Routing

Routing design involves the joint cooperation between an appropriate network routing algorithm and a switching structure to effect the steering of a transaction between subscribers. Circuit switch routing within the American Telephone and Telegraph (AT&T) system is based on a fixed nodal routing algorithm and a rigidly defined hierarchical structure in which a node higher in the hierarchy is not accessed unless blocking (no available circuit) occurs at a lower level. The Department of Defense (DoD) AUTOVON network is another symmetrically designed circuit switching structure so that all nodes are of equal rank. Furthermore, routing decisions in the AUTOVON are controlled at each node using an adaptive algorithm [24].

Store-and-forward routing algorithms are characterized by fixed or stochastic routing strategies with centralized or distributed control [43]. As opposed to the circuit switch routing process, a physical end-to-end path is not established. Instead, virtual circuit connections permit a transaction to proceed through the network with each node responsible for routing using nodal routing tables and knowledge of the system routing algorithm. The TYMNET and TELENET typify two of the most prominent commercial store-and-forward networks which exhibit differing routing design [77, 78].

In an integrated network any one or a combination of

routing algorithms may apply. However, a significant structural innovation occurs with the availability of the high speed, high capacity Common Channel Inter-switch Signalling (CCIS) system. CCIS in an integrated digital network can be obtained by reserving a specified number of frame slots for use as a signalling channel dedicated to the transfer of network signalling messages. Routing information is then transmitted over the channel by grouping routing data into multiple groups of Signal Units (SU), each 28 bits in length [11]. CCIS achieves error control by redundant coding within Signal Units and error correction by retransmission. The signalling network consists of common CCIS-equipped signalling capabilities at each node. Since the routing design must resolve problems involving speed, path selection, and the network decision making process, the innovation of CCIS demands a reappraisal of existing routing algorithms for use in an integrated environment.

2.1.4 Flow Control

A flow control strategy is concerned with insuring a smooth movement of traffic throughout the network under normal and/or adverse conditions. Flow control in data networks is frequently an application of queueing theory. The literature abounds with detailed models using queueing analyses for various configurations in telephone exchanges, computer networks, and host computer systems [6, 7, 26, 38, 49, 53, 64, 65, 67, 76]. The utility of data flow controls

must be subjected to close scrutiny under simulated integrated loading to determine any viability for use in a uniform network. Flow control link design must address problems associated with mixing classes of traffic over a common transmission medium. Frame slot sizes in allocation schemes are based on Voice Digitization Rates (VDR)'s which range from 16 Kb/sec to 64 Kb/sec for good speech characteristics, and packet sizes which may range to several thousand bits requiring multiple slots. Analysis of various models has resulted in several data/voice efficiency determinations [21, 25, 37, 57, 60, 84, 94].

The reduction in prices of vocoders (voice coder) and technical achievements involving packet speech transmission attest to the ready integration of voice and data [23, 25, 55]. Flow control design involving packet speech is primarily concerned with the network's ability to utilize free time in voice transmissions. By packetizing voice, it is treated similar to other data classes. If voice is not packetized, then other means must be employed to recognize periods of no data to fully utilize the link. Voice multiplexing is a technique that could be used to permit more efficient utilization of a voice link [23].

As previously described, network subscriber connections can be either physical or virtual store-and-forward circuits. The research model integrates both types. There is a consensus in the literature that both types of connections will

be needed in an integrated network [8, 21, 25, 41, 47, 52, 60, 79, 84, 99].

The packet design in an integrated network demands very responsive flow controls. The maintenance of proper flow rate between subscribers, and the prevention of nodal overloading are the two primary areas of control. Since a flow control design does not permit tractable analytical solution [30], simulation is a tool which will be applied to the research model to determine a strategy or combination of strategies which results in the most efficient system performance.

2.1.5 Nodal Design

Several network architectures have been proposed for the nodal integration of both voice and data [10, 25, 41, 47, 48, 54, 83, 84, 87]. Since the research model is based on an underlying circuit switch subnet with shared access to the subnet links, emphasis is restricted to the design considerations relating to this structure.

2.1.5.1 Circuit Switch Architecture

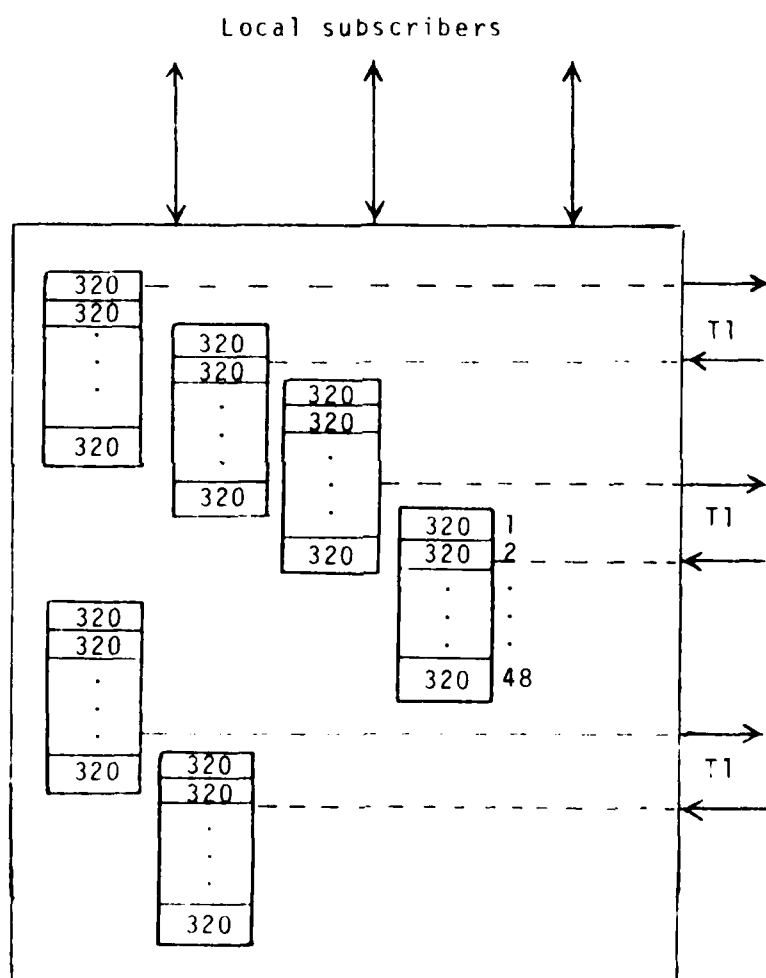
The recognition that modular multiprocessor configurations and Multiplexing/Demultiplexing (Mux/Demux) mechanisms are mandatory requirements [10, 41, 84, 99] is significant in the design of a circuit switch node. The termination of local subscribers, multiple T1 links, and requisite communications processing requirements dictate that a single CPU

architecture is inadequate. Fig. 3 illustrates a node terminating three FDX T1 circuits, with associated buffers. Assuming a VDR of 32 Kb/sec, a 10 ms frame period, all slots dedicated to voice, and a DMA of 320 bits per buffer, a processing capability of 28,000 interrupts per second would be required. Assuming a nominal processor capability of 500,000 instructions per second and an interrupt processing requirement of 20 instructions per interrupt, 576,000 instructions per second are needed, clearly beyond the capabilities of the node. Jenny [40] envisions a total workload on a nodal processor of 10,000,000 instructions per second. To satisfy this processing load Ross [84] provides an evaluation of alternative nodal architectures, recommending a distributed architecture with a partitioning of system processing to provide modular autonomy, parallel processing for high throughput, and inherent modular expandability.

The major software design required in each circuit switch node consists of (1) system routing, (2) communication processing, (3) line data Mux/Demux, and (4) data collection.

The system routing module is responsible for the selection of the most feasible path by coordinating the path selection with the Mux/Demux mechanism and the signalling network. When a route is requested, this module seeks a path based on the network routing algorithm using CCIS. Once a route is obtained, the control memory associated with the Mux/Demux mechanism is updated. The appropriate channel

Fig. 3. Circuit switch T1 terminations



tables must be updated to reflect a channel active condition along with any statistical updating. With the receipt of a connection/termination request, this module must coordinate the release of the Mux/Demux connection memory route, and perform any channel table updating.

The communication processing modules are responsible for the autonomous handling of the T1 links and any subscriber terminals. These modules must coordinate a Direct Memory Access (DMA) Input/Output scheme with the Mux/Demux mechanism and bus structure. All link and terminal error controls will be managed by this module.

The Mux/Demux mechanism utilized in an integrated network would improve on the digital crosspoint structured circuit switches. It is envisioned that larger high speed memories with stored program control and higher speed switching logic would be required [68, 93]. Several Mux/Demux designs have been proposed [10, 15, 20, 40, 41, 69, 90, 94, 99].

The data collection module would be responsible for gathering performance evaluation information. Essential statistics relating to link utilization, nodal traffic loadings, link and nodal failure occurrences, etc., are necessary to provide both grade-of-service and network tuning.

The storage requirements at each subnet node would consist of (1) channel buffers utilized for the time division

mechanism, (2) table storage, and (3) module storage.

Storage is not a major design consideration because minimal queueing occurs at subnet nodes.

2.1.5.2 Packet Switch Architecture

Each packet node terminates multiple types of data subscribers and interfaces to a circuit switch using a digital TDM link. Packet switch nodal design has three primary objectives:

- (1) To provide sufficient buffer storage for incoming/outgoing packets.
- (2) To insure the Multiplexing/Demultiplexing of subscriber processes according to a designated routing scheme.
- (3) To utilize flow control strategies which minimize queueing and congestion problems.

To accomplish these objectives, the nodal design must accommodate an order of magnitude increase in data traffic, and insure responsive flow controls and routing schemes. The termination of multiple subscriber devices ranging from 75 bps teletypes to 200 Kbs video terminals suggests a multiprocessor, multi-bus structure such as that employed in the PLURIBUS design [63]. Such a structure permits a flexible bus configuration of subscriber speeds. In addition, there is no assignment of priority among the multiprocessors, permitting autonomous processing of subscriber processes, with all processors having access to

system tables via shared memory.

The two major software modules required to support the stated objectives are (1) process Mux/Demux, and (2) process routing.

The process Mux/Demux module essentially establishes a logical connection (circuit) between any subscriber processes, and coordinates their arrival and their dispatch. The logical connection remains until the subscriber elects to break the connection. Subscriber connection tables must be maintained in each node to keep a record of any outstanding connections. In an integrated network, the problem of connection permission between classes becomes significant; that is, which subscribers are allowed to inter-communicate? The research model does not address communications between classes; however, module design could easily include such connections, using an added entry for permission authority.

The process routing module keeps a record of all current routes between this node and any other packet nodes. The destination number is the mechanism that accomplishes this routing, and at least two alternative techniques could be used:

- (1) Unique Global Destination Numbers
- (2) Unique Local Destination Numbers

Technique (1) uniquely identifies each process and node in the network. The obvious problem with this technique is one of sheer complexity and maintenance. A system overseer

must control the assignment of destination numbers.

Technique (2) proposes that each process header have an imbedded identifier which denotes the process as local or global to the packet switch. If identified as global, then a simple table look-up procedure can determine the destination number. At the destination end, the imbedded identifier can be similarly converted. A destination number identified as local to a packet switch is used for inward routing, i.e., local subscriber to local subscriber.

2.1.6 Priority/Preemption Controls

The requirement for network priority/preemption control is necessary in data networks [66, 90]. Design objectives are identified to be twofold:

- (1) There must be a mechanism that facilitates the rapid delivery of high precedence traffic, and
- (2) A capability must exist to preempt existing traffic connections.

The design of slot allocations must include priorities for subscriber traffic and probable system saturation conditions. Should the allocation scheme establish a priority for service, i.e., all voice calls first, then video, packets, bulk, etc? Should a prescribed number of slots be reserved for high priority traffic - independent of class? If there is no slot priority scheme, how are slots preempted in a saturated system? What selection criteria for slot removal is to be employed? How many

slots can be preempted? Should the priority/preemption structure be dynamically configured?

Precedence levels constitute a major nodal mechanism for providing grades of service [90]. A preemption capability can be implemented by permitting a high precedence subscriber transaction to interrupt an on-going lower level precedence transaction. Following the interrupt, resumption of the interrupted transaction can be reinstigated.

relationships involving routing and load sharing, efficient use of physical resources (buffers, transmission bandwidth, processor time), and an ever increasing requirement for support of diverse types of subscribers. The implementation or design of a viable flow regulated system and the creation of a desirable user environment involves the application of many communication and networking techniques. Major factors that impact directly on the need for flow controls are (1) synchronization of information flow on the data links, (2) detection and recovery from transmission error, (3) traffic routing from node to node, and (4) interfacing of subscriber processes to each other and the network [95]. The solution to the flow control problem has been ad hoc to date, but there is a growing consensus that it consists of a combination of flow control strategy and structural flow layers [30, 33, 35, 44, 45, 61, 62].

3.1 Structural Flow Control Layers

The Advance Research Project Agency's (ARPA) experience with the ARPANET has shown that structural layers of flow control, with each layer possessing well defined responsibilities, offers a unified networking implementation of the aforementioned flow control factors [33]. The international acceptance of X.25 (Interface Between Data Terminals Operating in the Packet Mode on Public Networks) [35] provides additional motivation for the layered approach. Although the formal layers of control vary from implementation to

CHAPTER III

FLOW CONTROLS IN DISTRIBUTED COMPUTER-COMMUNICATION NETWORKS

3. Problem Definition

The primary objective of all computer-communication networks is to transport information between its subscribers (typically a subscriber is a software processor, a subsystem, a human user, or a terminal device) in a timely and error free manner. Frequently, however, peak hour requirements, traffic load fluctuations, system component failures, or adverse network load distributions create an environment in which the movement of information throughout the network has to be controlled. Failure to do so results in congestion - a network condition as a result of which the network must reject any further input of traffic. Flow control is the set of mechanisms that attempt to match subscriber transmission and acceptance rates while minimizing network congestion. The development of adequate flow control mechanisms to regulate subscriber input rates within computer communication networks is generally referred to as flow control strategy [30].

The rapid growth in the last decade of distributed resource-sharing store-and-forward computer-communication networks such as ARPANET, TYMNET, TELENET, CYCLADES [33, 70, 92] has spawned a myriad of flow control problems. This has resulted because such networks possess complex inter-

implementation [1, 33, 72, 95], essentially all implementations provide three hierarchical levels of interface:

- (1) subscriber-subscriber
- (2) subscriber-subnet
- (3) node-node

Within each of these levels reside the software modules that produce the protocol needed to insure each subscriber-subscriber information transfer.

The subscriber-subscriber level performs as a software multiplexor/demultiplexor between subscriber processes and the interface to the subnet. Flow control at this level involves coordinating the dispatch and the arrival of information with subscriber processes and buffer storage [45].

The subscriber-subnet protocol coordinates the transfer of data between a subscriber process and the subnet. Flow control within this level is bounded by two extremes - a simple protocol that is applied only during the selection and the clear phases of a circuit or message (one or more packets) switch connection, to complex logic involving packet sequencing, efficient use of bandwidth, error detection, buffer allocation, subscriber responsiveness, and information transfer acknowledgement schemes using a High Level Data Link Control (HDLC) [35].

The node-node protocol level insures a smooth information flow from the time each message or packet enters

the subnet until it is delivered through an exit node. In all store and forward networks, this entails that an adequate supply of buffer storage exists along the path that the information will traverse. Flow controls supported by this protocol level are concerned with error detection and recovery capabilities, routing techniques (adaptive/deterministic), buffer availability, acknowledgement techniques, and the decoupling of the subscriber from any network responsibilities.

3.2 Overview of a Flow Control Strategy

A fundamental rule of communications systems is that the source subscriber cannot simply send information to a destination subscriber without some mechanism for guaranteeing storage for that data [56]. McQuillan and Walden suggest that this can be accomplished in two fundamental yet conflicting ways.

- (1) The source and destination subscribers agree on the amount of information to be transferred prior to the transfer, or
- (2) There is an acknowledgement sequence between source and destination subscribers throughout the information exchange.

Technique one is associated with high throughput (bulk type transfers), whereas technique two implies low delay (e.g., an interactive environment). All flow control strategies developed to date for use in store-and-forward systems have

contended with these conflicting techniques.

3.3 Objectives of a Flow Control Strategy

The goals and objectives of flow control strategies are frequently ill-defined but are generally categorized as follows [29, 71].

- (a) Congestion Prevention
- (b) Deadlock Prevention
- (c) End-to-End Flow Control
- (d) Fair Allocation of Communication Resources

3.3.1 Congestion Prevention

The flow control scheme should make it easy to slow or stop transmission from the source subscriber within limits compatible with available network resources.

3.3.2 Deadlock Prevention

When congestion does occur, resources must be available to handle traffic-clearing messages. In addition, to fully utilize the available bandwidth, different strategies may be appropriate for different modes of traffic.

3.3.3 End-to-End Flow Control

Flow control for a particular level of protocol should be exerted at the point closest to the final destination. In addition, the flow control strategy must consider the resource management scheme provided by a destination subscriber or exit node.

3.3.4 Fair Allocation of Communication Resources

A flow control strategy must insure that resources such as nodal and subscriber buffers and communication bandwidth are not monopolized by one source/destination subscriber or that component failures do not invalidate the strategy.

These flow control strategy objectives are difficult to achieve, and are further complicated by the addition of control information overhead and the quest for network transparency and resource optimization.

3.4 Flow Control Strategies

Although flow control strategies are classified in the literature as local or global [6, 30, 45], a more meaningful network relationship can be discussed by considering whether or not the strategy is subnet or subscriber controlling. That is, does the strategy attempt to regulate the network directly or the subscriber? Existing or proposed strategies that apply as potential techniques for an integrated network (Table I) will now be discussed.

3.4.1 Subnet Controlling

Subnet controlling strategies attempt to regulate the subnet nodes, thus preventing the network from becoming overloaded. This is accomplished either through the use of table and control information periodically being passed through the network, or by the detection and correction of a congestion in a particular node by a centralized

Table I. Classification of Flow Control Strategies

<ul style="list-style-type: none">. Subnet Controlling<ul style="list-style-type: none">. Isarithmic. Centralized Flow Control Station. Flow Control Tables. Nodal Buffer Monopoly. Overflow Detection. Nodal Limits. Node-Node ARQ. Dedicated Nodal Buffers
<ul style="list-style-type: none">. Subscriber Controlling<ul style="list-style-type: none">. Reservation Schemes<ul style="list-style-type: none">. RENM. Windowing. Crate Occupancy. Subscriber-Subscriber<ul style="list-style-type: none">. Circuit Establishment. Constant Rate. Pause and Go

intelligence center or autonomous nodal design.

3.4.1.1 Isarithmic

This strategy, first proposed by Davies [12], is based on the premise that network congestion can be measured by the number of packets (a packet is a fixed number of bits) in transit over all communication paths in the network. With this approach the information transfer can be regulated by placing a limit on the total number of packets in the network at any time t . Simulation studies [12, 73] have modeled this approach by associating at each node two queues - a permit and a packet queue. As each packet enters the network, the source node permit queue is decremented by 1; likewise, on arrival at a destination node, the permit queue is incremented by 1.

Simulation performance results and improvements to the model involving permit redistribution schemes have shown the isarithmic technique to be an effective mechanism for regulating input traffic operating under a variety of control protocols and system loads. Although this approach has not been implemented, it does highlight the dilemma between high bandwidth and low delay (computationally simple), yet appearing to be stable for large networks [12, 33]. It has three major drawbacks, namely:

(1) There are considerable delays before some packets receive attention due to "bunching" of permits at some nodes when needed elsewhere, (2) it does not correct the congestion

problem when a destination subscriber cannot accept packets fast enough from a source [33], and (3) it has no method for redistributing permits when network component failure has occurred.

3.4.1.2 Centralized Flow Control Station

A centralized flow control strategy is one which utilizes an intelligent control point for monitoring network performance. This control mechanism (a master supervisor) is constantly exchanging control information with all subnet nodes to determine delays on all paths. The global status of the network is maintained through a flow control matrix whose elements represent the delays received from the nodes on a periodic basis.

Each node contributes a column specifying the delays from that node to its directly connected neighboring nodes. For example, this strategy is implemented on the TYMNET network by a supervisor module in the Network Routing Center (NRC), which establishes a route for each transaction between a source-destination pair using the delay matrix information [86, 92]. This strategy provides global knowledge for routing and control [86]. It has major shortcomings in its inherent difficulty of reacting quickly to nodes remotely located from the control mechanism and its heavy utilization of network bandwidth for monitoring and control [33].

3.4.1.3 Flow Control Tables

Gerla described a flow control table strategy where each node controls its input rate on the strength of information contained in flow control tables which are periodically circulated in the network [30]. These tables, continuously updated while circulating in the network, contain information regarding buffer availability in each node. The information at a specific destination allows a source node to control the input-rate directed to the destination node. The sum of the buffers available over all nodes provides an indication of network congestion. In addition to the overhead of passing flow control tables, this strategy reflects a buffer availability at some previous time, thus limiting the effectiveness of the tables. However, as in the centralized controller strategy, total system visibility is obtained.

Gerla proposed a global windowing scheme as an improvement to the previous method, in which each source node has a pool of numbers to each destination (or group of destinations in the same region or subnetwork). Each packet sent is assigned a free pool number, and this pool number is returned to the pool once receipt of an acknowledgement from the destination node has been obtained. The size of the pool is variable and can be adjusted on the basis of flow control table information. Unfortunately, this improvement is limited in its effectiveness because it relies upon an acknowledgement which may not be forthcoming due to a

destination subscriber process problem.

3.4.1.4 Nodal Buffer Monopoly

This flow control strategy attempts to exploit the relationships between traffic flow on input and output ports (channels) at each node [42], given the equilibrium transmission state at a node, for which Σ input traffic (I) = Σ output traffic (O). Due to nodal buffering (B) and traffic flow fluctuations, this equilibrium condition is seldom true. However, it is true that

$$\Sigma_i \int_0^t I_i(t) dt - \Sigma_j \int_0^t O_j(t) dt \leq B \quad (3.1)$$

that is, at any point in time all information can be accounted for on input/output ports i and j respectively.

The state of a node can be represented in terms of the information volume transferred via an input port and its load on B until it has been transmitted out over the various ports j. A matrix S (Fig. 4) can be used to represent the state where each S_{ij} is the load on B caused by information flowing internally from port i to port j at time t. Thus equation 3.1 can be re-expressed as

$$\Sigma_{ij} S_{ij}(t) \leq B, \text{ in which} \quad (3.2)$$

any row vector $R_i(t) = (S_{i1}(t), S_{i2}(t), \dots, S_{in}(t))$

represents the instantaneous load on B contributed by input port i. Any column vector

Fig. 4. Buffer state matrix

		Output Ports					
		1	2	.	.	.	n
Input Ports	1	S_{11}	S_{12}	.	.	.	S_{1n}
	2	S_{21}	S_{22}	.	.	.	S_{2n}

	n	S_{n1}	S_{n2}	.	.	.	S_{nn}

Monopoly Matrix S

Each S_{ij} entry represents the buffer loading from input port i to output port j

$$C_j(t) = (S_{1j}(t), S_{2j}(t) \dots S_{nj}(t))$$

represents the instantaneous information volume waiting at port j to be transmitted and the loading on B while waiting. With this knowledge it is possible to observe several different types of buffer monopolization. For example, as vector $C_j(t)$ approaches B , an output port is causing a bottleneck. As $R_i(t)$ approaches B , an input line can be found to be the cause of congestion. One port/port information path can be found to monopolize B as $S_{ij}(t)$ approaches B .

Buffer monopolization controls for each of these situations can be determined by setting limits (threshold values) for the vector entries $R_i(t)$, $C_j(t)$ and S_{ij} . The primary advantages of this strategy consists of its ability to (1) push back deviations as they are detected to the source, (2) detect and identify several causes of nodal congestion, and (3) eliminate the need for control information overhead. However, this strategy does suffer from frequency of update overhead.

3.4.1.5 Overflow Detection

Kahn and Crowther [45] describe a flow mechanism, used in the ARPANET, which insures that a small fraction of subscriber traffic is always capable of being delivered. The method centers on two overflow buffers maintained in each Interface Message Processor (IMP). When information flow

is at a virtual standstill, an IMP marks a packet for delivery, starting the overflow process. The overflow packet passes from IMP to IMP using the overflow buffers in each IMP until it reaches its destination. This strategy is especially useful when a loop exists where each IMP is saturated with traffic bound for another IMP in the loop.

3.4.1.6 Nodal Limits

This strategy is prevalent in message based store-and-forward systems such as AUTODIN (Department of Defense Automatic Digital Network). Input traffic is permitted to enter the system until a preset storage threshold is reached. Then a Wait Before Transmit (WBT) communications protocol character locks out any further input of local subscriber traffic until the traffic load subsides. This strategy is computationally simple to implement, but does not necessarily impact the source of the congestion, since neighboring nodes are not made aware of the problem.

3.4.1.7 Node-Node ARQ (Automatic Request for Repeat)

Various node-node ARQ schemes are used in store-and-forward networks to insure the timely and error free flow from subnet node to node. They are based on packet/message acknowledgement (ack) techniques such as Stop-And-Wait ARQ, Continuous ARQ, Go-Back-N, Selective Retransmission, Adaptive ARQ, etc [14]. The proper receipt of an ack permits the sending node to free up space held for the message or packet

and assigns delivery responsibility to the on-going node. To minimize the delay incurred while waiting for an ack, several techniques have been implemented to better utilize the bandwidth. Paramount among these are

- (1) "piggybacking" acks on FDX links,
- (2) using one ack for a group of packets or message segments,
- (3) responding with negative-acks (NAK)'s only.

3.4.1.8 Dedicated Nodal Buffers

A nodal dedicated buffer strategy has been implemented in the ARPANET to preclude direct and indirect store-and-forward lockups [33]. Direct store-and-forward lockups occur when two IMPs are each saturated with packets bound for each other. An indirect store-and-forward lockup involves a loop of IMPs such that each is saturated with packets for another IMP in the cycle. This strategy utilizes buffers that are reserved for the input/output of traffic. A reserved output buffer guarantees that output to another node is always possible, while the reserved input buffers permit each node to always be able to inspect incoming packets. Although this strategy was contrived to correct a most serious ARPANET problem, the strategy also has utility for whenever these types of lockups occur.

3.4.2 Subscriber Controlling

Subscriber controlling flow control strategies are

characterized as end-to-end strategies, where the end points are subscribers or the entry/exit servicing subnet nodes.

3.4.2.1 Reservation Schemes

The Request For Next Message (RFNM), an ARPANET originated strategy, permits only one message at any one time in transit through a logical (space) subscriber-subscriber connection. This strategy became necessary to prevent a "reassembly lockup" condition at the destination node [45]. Reassembly lockup arises when a number of partially reassembled messages (multiple packets) have occupied space in a destination IMP. Space cannot be released for new packets until a message has been assembled and delivered to the subscriber. This, however, is not possible because the packets needed are held up in the network waiting for reassembly space in the destination IMP. The use of reassembly buffers as a mechanism to regulate the flow between two subscribers requires coordination between the source and destination IMP to insure sufficient buffers are available for this transaction. This coordination on a per message basis assures that a single link cannot cause network congestion. The source subscriber cannot send another message to that destination over that logical link until an RFNM is received. The actual connections are functionally like circuit switch connections, but in fact take the form of table entries, used to create virtual circuits from one subscriber to another.

The National Physical Laboratory (NPL) uses a flow control strategy logically similar to the RFNM reservation, yet implemented in quite a different manner [13]. When two subscriber processes in the NPL network wish to communicate, a FDX call is established. Unlike ARPANET, calls are set up using special message types (16 possible). Flow control is achieved by requiring that messages be sent only if a go ahead message type has been received by the source subscriber. Unlike ARPANET, several messages may be in transit.

In summary, an RFNM type strategy does regulate flow control and prevent reassembly lockup, but at the expense of bandwidth reduction. Additionally, end point subscribers must maintain all information regarding the connection to insure smooth flow control.

A window flow control strategy, first proposed by Cerf and Kahn [5], can be used in conjunction with a source subscriber retransmission capability to provide considerable flexibility for buffer allocation schemes. Basically, each character or packet has associated with it a unique sequence number which is its byte location relative to the beginning of a data stream. Assume that this sequence number is modulo n . A flow control window is a dynamic range within this sequence number space. The window edges define the window range. After both subscribers are synchronized (similar left edges), the left edge of the window plus the

window size give the highest sequence number (w) that can be transmitted ($0 < w < (n-1)$). With this strategy the receiver is free to vary the window size, according to any desired algorithm as long as the window size never exceeds a threshold which would cause the receiver to think that he is receiving a new message [5]. As long as arriving packets are within the window range, the receiver will return to the sender the next expected sequence number. The primary advantage of the window strategy is that it can be used by both source and destination as a flexible and robust tool for tuning the data flow over a connection. The source can "push" the receiver by sending packets at a high rate, subsequently adjusting the window to the rate at which acknowledgements are received [2]. The destination on the other hand, may want to alter the window size to reduce the local buffer load. The destination may discard a packet, knowing the source will time-out and retransmit, thus eliminating out-of-sequence problems. A major requirement to implement this strategy is a source retransmission capability.

3.4.2.2 Crate Occupancy

A pair of subscribers that desire to communicate could each at the outset have provided a prescribed number of buffers. These buffers are exchanged using the mechanism of crates [45]. When a crate arrives at one subscriber, it is used to return a message. As long as crates are

available, flow can continue. Thus crate availability regulates traffic flow. Although this method is computationally simple, it controls congestion on a subscriber-subscriber basis only, and does not insure that crates are available where and when needed.

3.4.2.3 Subscriber-Subscriber

Subscriber-subscriber flow controls are autonomous from subnet functions. Once a physical connection is established through the network, the subnet appears to a subscriber as a physical private circuit. All communications between subscribers require some knowledge of capabilities existing at each end prior to connection establishment. Subscriber-subscriber flow control strategies have traditionally been used primarily in circuit switch networks. Some store-and-forward switched networks possess a circuit-switch capability [85, 92] with limited information exchanges permitted between network types. Data flow control mechanisms in use to date have been ad hoc designs developed for use between subscribers with common interests. Each of these mechanisms support to varying degrees the major responsibilities of error and delivery controls, data link controls, speed conversion, data conversion, record keeping and mode of operation (half/full duplex). Although the advent of the microprocessor provides increased flow control capabilities, the implementations to date of subscriber-subscriber flow mechanisms fall into three

categories:

- (1) Circuit Establishment
- (2) Constant Rate
- (3) Pause and Go

Circuit establishment mechanisms are prevalent in a dial-up mode of operation. Each subscriber must have some minimal knowledge of the device characteristics at the destination end. Typically, the subscriber can select the baud rate, mode of operation (half/full duplex), and parity. Once a connection is made in dial-up mode, the subscribers are free to communicate with no other controls applied. Typical applications of this mechanism are teletypewriter-teletypewriter dialogues, time-sharing operations, and facsimile transmissions, where higher error rates can be tolerated. Since the circuit is dedicated for the duration, bandwidth utilization is extremely poor. However, this technique is very economical for remote user support in time sharing operations.

Typical applications of the constant rate scheme involve the transmission at a constant speed over a circuit switch connection to an output peripheral from some intelligent device (i.e., computer or communications controller to a teletype like device or printer) [74]. Since the peripherals contain minimal logic and receive only at a hardwired rate, they cannot influence the sender. Some error flow controls are applied to signal abnormal conditions (paper shortage,

line loss, etc.). For the most part, however, the output peripherals rely on the sending device for any flow controls or record keeping.

Pause and go is the principle technique used between subscribers, when accuracy and transmission controls are overriding considerations. With the wait and go technique, the sender must pause after transmitting a prescribed number of data characters until receipt of a go ahead control character. This process is then repeated [100]. Typical applications of this technique can be observed in data base updates from a remote terminal, or a magnetic tape-magnetic tape data exchange. Since accuracy is essential for these operations, either source or destination can initiate wait commands at any time. All data storage and record keeping may be uniquely performed by either subscriber since no flow controls govern a common responsibility.

3.5 Conclusions

The need for flow controls has developed as a result of the growth in resource-sharing store-and-forward based networks. Flow control mechanism developments have been generally ad hoc designed and tailored for specific implementations. A review of what has been implemented or proposed enabled classification of existing mechanisms into subnet and subscriber controlling. With the advent of the all digital network, and probable integration of diverse subscriber types, flow controls have/and will become

increasingly complex. The nodal buffer monopoly and windowing schemes intuitively appear to jointly provide the flexibility and robustness needed for use in an integrated network and will be examined more closely in the network model. Modern micro-technology development makes subscriber-subscriber flow control strategies cost effective as an alternative to subnet overhead processing.

CHAPTER IV

THE NETWORK SIMULATION MODEL

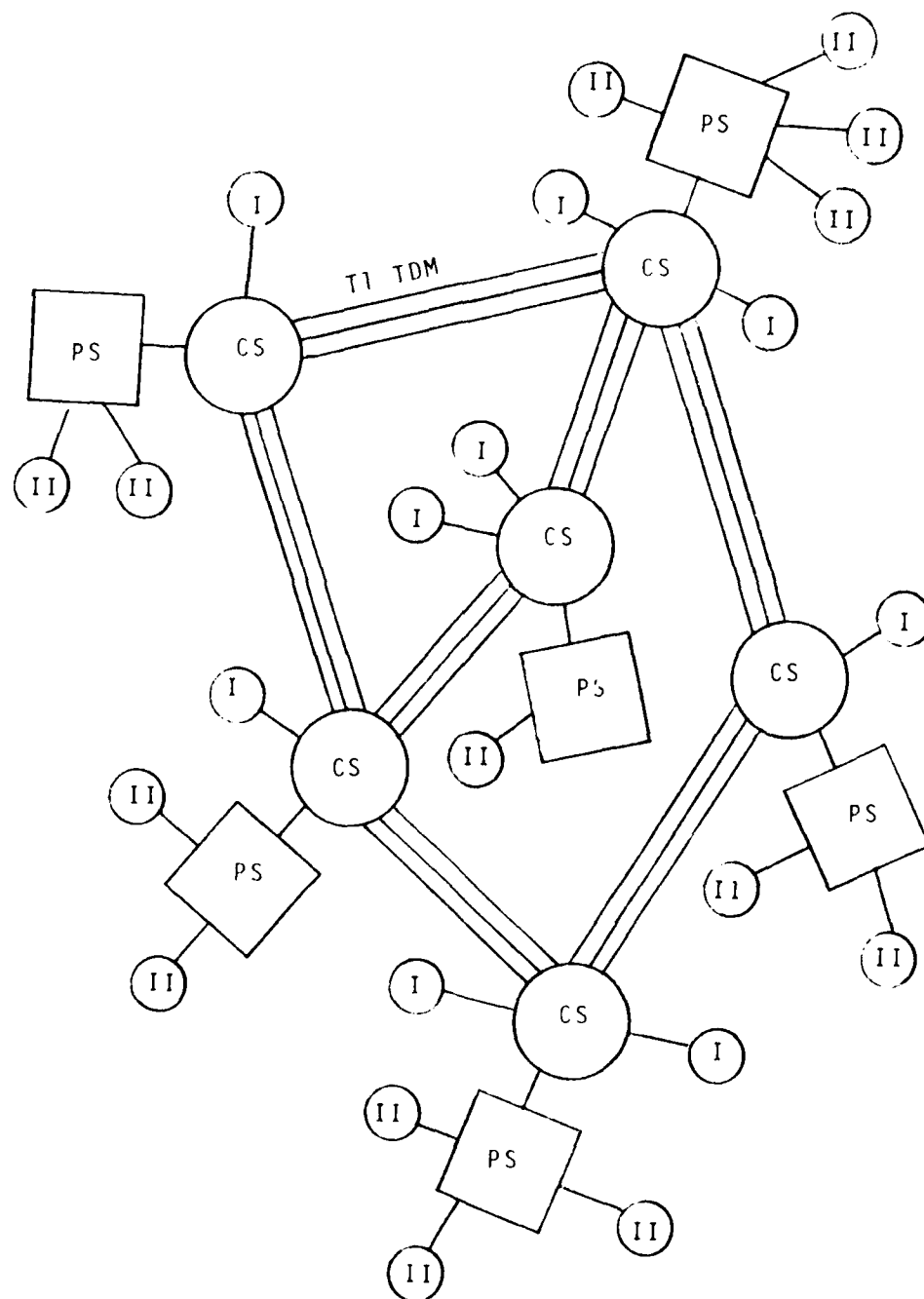
4. Introduction

The model developed in the simulation program is based on an integrated packet/circuit switched network consisting of the following major components: (Fig. 5)

- A. Backbone Circuit Switch (CS) Nodes
- B. Peripheral Packet Switches (PS)
- C. Invariant Network Synchronous Time-Division-Multiplexed (TDM) Frame Switching Superstructure
- D. Digital Network Using T1 Carriers and Digital Switching Nodes
- E. Variable Subscriber Data Rates
- F. Two Classes of Subscriber Traffic
 - 1. Class I - Real-time traffic that once started cannot be interrupted (voice, video, facsimile, and sensor)
 - 2. Class II - The general class of store-and-forward (packet based) data, such as interactive, query/response, bulk, and narrative/message

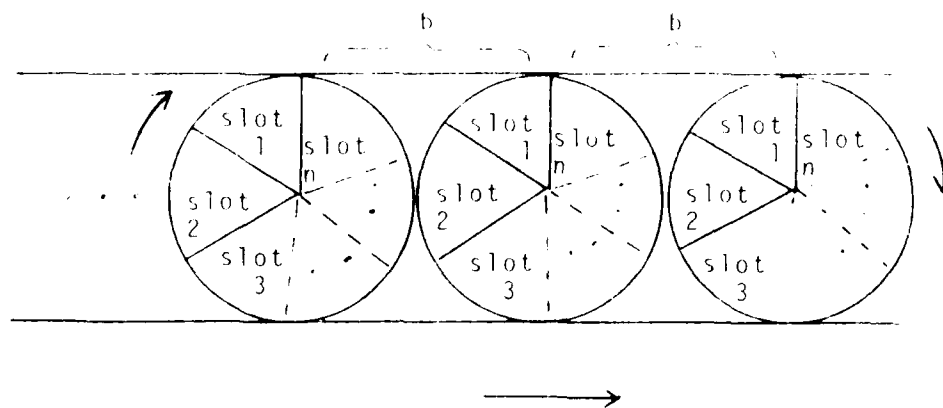
The backbone CS network nodes and peripheral packet switches form the nucleus for a distributed computer-communication network in which the transmission of data/voice between any

Fig. 5. Network configuration



two subscribers is accomplished via adaptive sharing of the high capacity T1 link bandwidth using the concept of the SENET (Slotted Envelope Network) [10] self synchronizing switching superstructure. This concept treats the available bandwidth on a digital link as a resource for which all forms of communication must compete. Using SENET the T1 link is synchronously clocked into frames of a time duration which are assumed invariant throughout the network (Fig. 6). Using a competitive allocation methodology, each frame is structured to encompass a diversity of trunk communication rates in order to serve a multiplicity of diverse Class I/II subscriber traffic. The allocation implementation partitions the frame into several data slots (channels of variable time duration). The self synchronizing capability within each frame is implemented by assuming a Start-of-Frame (SOF) marker (a few bits at the start of each frame) to indicate the beginning of each of a contiguous series of constant period frames. Following this marker, the remainder of the frame is divided into Class I/II slots according to the grade-of-service designed into the system. Class I slots contain those types of traffic that are normally associated with circuit switches. The Class I slots provide physical channel connections that are both dynamically allocated and maintained in accordance with routing tables at source, intermediate, and destination nodal switches. Changes in these routing tables are coordinated using a routing module

Fig. 6. Master frame clocking



that assumes a signalling capability exists. Class I subscribers are directly terminated to circuit switches to preclude packetizing and any unnecessary routing overhead through packet switches. Each dial-up Class I connection results in a physical subscriber-subscriber connection for the duration of the call, or a system "loss", similar to a telephone dial-up process.

Co-located with the circuit switch nodes (although not a design requirement) are packet switches, terminating all Class II subscribers. The transmission of data between the packet and circuit switches is accomplished using Time-Division-Multiplexing on the network side, while the packet switch/subscriber interface is dependent upon the individual terminal hardware configurations.

The packet switches are primarily responsible for management of packets between input terminals and the circuit switches, placing traffic on queues according to a regional routing policy, and performing connection initiation, circuit disconnect, and coordination with other packet switches depending on the system loading.

Traffic flow control mechanisms are used between either adjacent nodes or on an end-to-end basis in existing packet-switched based networks. Using the packet as a measure of congestion, the developed overhead as a result of flow controls severely restricts the peak bandwidth that is available to individual users [30, 45].

The proposed regional routing doctrine for each packet switch coupled with virtual switch connections, reduces this overhead and the congestion problem. As traffic is entered into a packet switch from subscriber terminals, it is queued for the relevant destination packet switch. Unlike the SENET scheme, a circuit switch connection is then initiated/terminated by the packet switch on behalf of this traffic. A circuit switch connection can be established for a single transaction similar to an interactive communication, or on a multiple transaction basis if the traffic is bulk data, message/narrative traffic, or several users queued for the same destination packet switch. This routing scheme (1) insures minimal queue build-up within the backbone, (2) enforces an end-to-end flow control strategy, and (3) requires responsive link flow controls.

Deterministic or adaptive routing schemes that are currently used in existing packet/circuit switched networks or hybrid combinations could have been employed within the backbone network [36, 87]. Progressive alternate routing is used in this research model. With this method each circuit switch node has a primary and an alternate path. If blocking occurs at some node during connection initiation, the alternate route is tried for route completion. If this connection fails, the transaction is either queued at the packet node or considered a system loss at the circuit node, depending on its class.

The primary goal of this research model is to provide a tool which can be used to determine the system flow controls which reduce end-to-end loss probabilities for Class I traffic and end-to-end packet delays and queues for packet switched traffic, given variable link capacities, dynamic frame slots, varying packet sizes and Voice Digitization Rates (VDR).

4.1 Description of the Queueing Model

Whenever applicable, a simple analytic model that yields precise algebraic expressions relating system inputs and outputs is preferable over any corresponding simulation model. For the integrated computer-communication network previously described, the numerous inter-nodal conditions and variables preclude any exact analytic solution. However, by decomposing the network into nodal queueing models, a significant objective is the verification that the research model output can be represented with simpler models.

The traffic flow at each packet switch is described as follows:

1. Each Class II subscriber communicates with the packet switch via independent, Poisson transaction arrivals and exponentially distributed transaction interarrival times.
2. The message lengths (packets per transaction) are assumed to be geometrically distributed. This conforms to the study of multiaccess computer

communications by Fuchs and Jackson [26].

3. Statements (1) and (2) imply that transactions arrive at the packet switch independently according to an exponential interarrival distribution [64].
4. The number of packets that arrive at the packet switch between (any) two completed transactions is a random variable with geometric distribution.
5. Each packet switch can be thought of as a M/M/C system (Kendall notation) [32], with infinite storage.
6. Packets are placed on the packet switch queue and served on a first-come-first-served (FCFS) basis.

The traffic flow entering each circuit switch node originates at neighboring circuit switch nodes, connected packet switch nodes, or locally terminated Class I subscribers. Since all traffic entering from other than terminated subscribers see a physical connection, it does not enter into a serving mechanism process at the circuit node. The locally circuit switch terminated subscribers are assumed to possess Poisson arrival and exponential service distributions. Thus the M/M/C/C queueing model suffices to represent this network model.

Both queueing models are impacted by channel availability. To accomplish the time slotting of the frame at each circuit switch node, the literature [10, 21, 60, 83] promotes the use of an operating rule that allows subscribers

access to the system using a "gate" concept demand from queueing theory [95]. The gate is assumed to be open at certain time intervals to Class I or Class II data. This is done to insure a minimum grade of service. For the research model, the frame boundary is "floated", thereby keeping the gate open all the time. This forces Class I and II subscribers to compete for available slots.

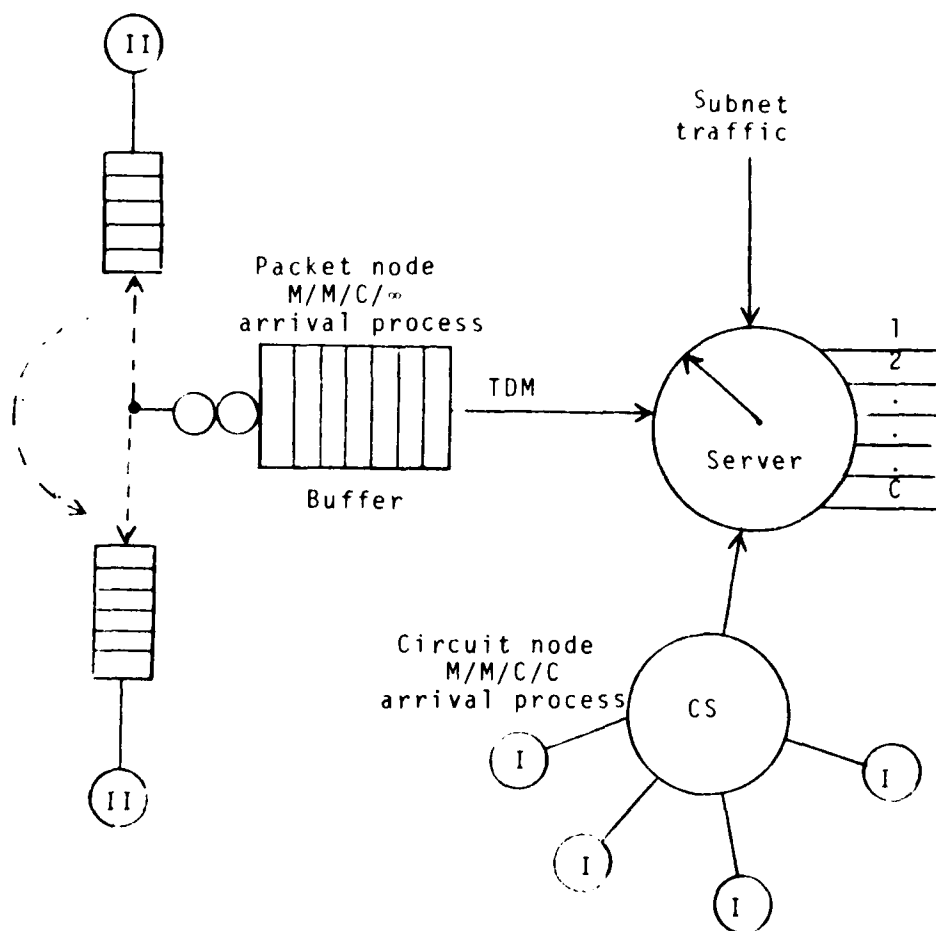
In summary, delays and queues at each node are approximated closely by $M/M/C/\infty$ and $M/M/C/C$ queueing models (Fig. 7). Since each of these models is globally impacted by channel availability, the network simulation provides performance measures for end-to-end delay and blocking with relevant flow parameters applied.

4.2 Overview of the Simulator

The creation of a program model implies a two-step procedure of (1) detailing a logical description of the system to be modeled and (2) programming and testing that model.

The logical description of the program model is closely related to the usage of a symbolic language to give a compressed yet correct description of the studied system. Since the construction of the appropriate model is an essential part of systems analysis, it is useful to distinguish between the different types of models available to the analyst. Emshoff [17] classifies models into four major types (descriptive, physical, symbolic, and procedural)

Fig. 7. Network nodal model



comparing their usefulness with such factors as ease of communications between technical and non-technical personnel, the relative cost of using such models, and the associated limitations for each. Of these four types, the symbolic and procedural models offer the greatest potential for data communications oriented problems. A symbolic (mathematical) model is an efficient tool where explicitly stated concise mathematical expressions can be formulated. For those data communications systems which are event driven by real-time requirements, it is difficult to quantify, using analytical methods, the dynamic relationships which exist among numerous system variables. Instead, the use of procedural (simulation) models becomes necessary to express the dynamic relationships hypothesized to exist in the real world situation, by means of a series of elementary operations on appropriate variables. These elementary operations are typically represented by use of a computer program.

Formal computer languages have many characteristics. The relative importance of these characteristics depend on such factors as availability, ease of programming, diagnostic capability, applicability to a wide range of problems, and one that facilitates model formulation. Although there are a number of high level program using languages especially designed for discrete system simulation, such as GPSS, GASP, SQL, SIMSCRIPT [17], the FORTRAN language was selected for development of the model because in addition to possessing

these features, it was readily available on both the local Amdahl 470 and the Data General Nova Series computers as well as other minicomputers.

Since the communications activity was centered around nodal activity, the model was node based. There are three principle nodal tables - routing, channel, and queue tables. The routing table is used to determine the output channel a transaction will take through the node to reach its destination. This table is created in the initialization phase of the program. Each channel table is sized for maximum voice channel allocations. Varying a system parameter increases/decreases these allocations. Since each connection is FDX, two indices, representing sender and receiver, are required, and each connection is modeled by two independent channels. Information stored and updated within various channel table entries at each node is used in the gathering of statistics. The queue tables at each node are used to obtain statistics associated with average transaction time in the system, utilization of the channels, and various transaction oriented delay statistics.

An event table reflects network status disturbing events. For example, an event table entry may contain the time of the next arrival or departure at a node. The simulation is driven by event changes that occur at each node.

4.3 Properties of the Model

The model is partitioned into functional modules. Each such module consists of a number of event routines or subroutines. The driver module is scheduled from input parameters. All other routines are scheduled by the research model itself.

4.3.1 Operating Characteristics

For a 10-node network the compiled object program occupies 30K bytes and the associated table and work areas require storage based on network sizing. The computer running time is simulated by nodal event occurrences for a period of time as specified by the user input system parameters during the initialization phase. Steady state is assumed when a node reaches a user provided packet saturation level or end run time. At this time channel and queue statistics are collected. Although each computer run is based on a user supplied end time, it was felt that several minutes (8) of nodal simulation permitted sufficient statistics to be obtained. The main program consists of a tight DO Loop, calling the event module until the end time is reached, at which time output statistics are printed, followed by run termination.

4.3.2 Description of Functional Modules

The major modules of the research model are:

- (1) Initialization

- (2) Transaction Generation
- (3) Arrival
- (4) Routing
- (5) Departure
- (6) Event Selection
- (7) Statistical Gathering and Output

4.3.2.1 Initialization

This module has a twofold objective: (1) to build tables according to user input parameters, and (2) to initialize table areas at the beginning of the run and initialize several table statistic entries when steady-state is reached. These objectives are accomplished within the NEWMSG subroutine and the READ statements of the main program.

4.3.2.2 Transaction Generation

This module is comprised of subroutines NEWMSG, POISSON, GEOM, and RANDOM. Each call to the generation module results in an arrival containing the length, destination number, and arrival and departure times of a transaction. The arrival information is then placed in an available queue location.

4.3.2.3 Arrival

The arrival of a transaction at a node is a status disturbing event. The arrival module is responsible for:

- (1) Determining whether the new arrival requires a route or can be placed directly on an existing

channel queue.

- (2) Initiating a route through the network, if required.
- (3) Updating channel routing tables at all affected nodes.
- (4) Calculating the time this transaction would depart the system and recording this in all affected channel tables.
- (5) Determining queueing delays and updating nodal delay tables.
- (6) Generating the next arrival at this node.
- (7) Updating the system clock.

Subroutines required to accomplish these goals are ARRIVE, IMPROV, and UPDATE.

4.3.2.4 Routing

This module is responsible for initiating and terminating a FDX route from source to destination through the network. Passed to this module are source and destination arguments. Since the ROUTE subroutine has knowledge of end points, it searches routing tables in a prescribed manner to derive a route. The variable IYESNO indicates the status of the route search. Once a path is found through the network (IYESNO=0), routine UPDATE is called to actually construct the path and update channel table entries at affected nodes. If no path can be found (IYESNO=1), a failure indication is returned to the calling module.

Using source and destination nodes, the output channel at each node is identified via a CHANIB table look-up. Each intermediate node is then accessed using the appropriate output channel. In this manner each node is accessed and channel and queue purging accomplished.

4.3.2.5 Departure

The departure of a transaction from a node is a status disturbing event. The departure module has the following primary responsibilities:

- (1) Terminating the route through the network.
- (2) Updating channel table and queue entries for each affected node.
- (3) Updating the system clock.

The subroutines required to perform these departure functions are DEPART and REMOVE.

4.3.2.6 Event Selection

The event selection module is responsible for (1) obtaining the next event at a node and placing it in the node event table, and (2) selecting the next event and calling the appropriate servicing module. Nodal arrival and departure queues are searched for the earliest event occurrence by subroutine NUIVNI. Information related to the event is then placed in the event table.

The event table is scanned by subroutine EVENT for the next event in time to be serviced and a branch made to the

appropriate servicing module using a COMPUTED GOTO statement.

4.3.2.7 Statistical Gathering and Output

This module uses the accumulated channel and queue data to output statistics relating to channel and node usage. Included in the statistical output are a summary of traffic flow through each node, including channel utilization, packet delays, voice blocking, slot utilization, and nodal loading. Subroutine STATS provides the output information while subroutines UPDATE, ARRIVE, DEPART, and IMPROV update channel and queue tables.

4.4 Description of Major Tables

The following tables are necessary to integrate the functional modules:

1. Parameter (PARAM) [X]
2. Event (EVENT) [Node, Entry]
3. Destination (DESTAB) [Node, Dest]
4. Channel (CHANIB) [Channel, Entry]
5. Queue (QUEUE) [Node, Entry]
6. Call Queue (CALLQ) [Node, Entry]
7. Cumulative Time (CUMTIME) [Node, Entry]
8. Calls Accepted/Rejected (CALLS) [Node, Entry]
9. Link (LINKIB) [Node, Dest]
10. Seed (SEEDIB) [Node, Distribution]
11. Line Availability (LINES) [Channel]
12. Queue Entry Count (QENT) [Node]

13. Channel Connectivity (CDDCHL) [Channel]

14. Alternate Channel (ALICH) [Channel]

15. Circuit Switch Arrival (CSARV) [Node, Entry]

16. Alternate Destination (DSTAL?) [Node, Dest]

Tables A1-1 - A1-16 contain a description of each one.

CHAPTER V

ANALYTIC DEVELOPMENT OF AN INTEGRATED MODEL

5. Introduction

The integrated network can be decomposed into multiple nodal configurations, each comprised of two exponential arrival processes competing for a fixed number of common time slot allocations (channels). The arrival process to the channels is a Poisson stream of transactions with mean arrival rates λ_1 and λ_2 respectively. Similarly, the service times are mutually independent, negative exponentially distributed random variables with mean service rates μ_1 and $\mu_2 > 0$ respectively. A Class I transaction that finds all channels busy departs the system and is considered a system loss. A Class II transaction that finds all channels busy joins the queue and remains there until served.

Each nodal system is modeled as a Markov process with state space E , where E is defined by

$$E = \{(n,m): n = 0,1, \dots, C; m = 0,1, \dots, \infty\}$$

The steady-state probability that m Class I transactions and n Class II transactions are in the system (the system consists of all channels and the queue of Class II transactions, if any) is denoted by $p_{n,m}$. For notational simplicity the vector of steady-state probabilities, P , is in lexicographic order and is partitioned as (P_0, P_1, \dots) ,

where $P_j = (p_{j0}, p_{j1}, \dots, p_{jC})$ for $j = 0, 1, \dots$

Using the traditional methods of Markov processes [32], the steady-state probabilities are the solution to

$$Pr = 0$$

$$\text{and } \sum_{i=0}^{\infty} \sum_{j=0}^C p_{ij} = 1,$$

where P is the matrix of transition rates. Fig. 8 depicts a rate matrix for a three channel system.

The rate matrix P can be structured as a block tri-diagonal form. Wong, Disney and Giffin [96] developed a solution technique for a queueing system whose rate matrix has a similar block tri-diagonal form. Their work is not directly applicable to the nodal model for two reasons:

- (1) They assumed a finite queueing capacity, and
- (2) Their solution depended on the sub-diagonal partitioned matrix being non-singular.

In contrast, the nodal model allows data queues to form when all channels are filled with voice or data. Additionally, it is necessary to rotate the last column of each channel group circularly to the left one group to preclude singular sub-diagonal partitioned matrices.

The goal of the algorithm developed herein is to obtain steady-state probabilities for the number of Class I/II transactions within a nodal system. A large rate matrix, $(C+1)^{**}L + (C+1)^{**}L$, $C \geq 48$ and upwards, is required to obtain steady-state solution for the nodal model. Clearly, for

	channel group 0				channel group 1				channel group 2				channel group 3				channel group 4			
nm	00	01	02	03	10	11	12	13	20	21	22	23	30	31	32	33	40	41	42	43
00		λ_2			λ_1															
01	μ_2		λ_2			λ_1														
02		$2\mu_2$		λ_2			λ_1													
03			$3\mu_2$					λ_1												
10	μ_1					λ_2			λ_1											
11		μ_1				μ_2	λ_2			λ_1										
12			μ_1			$2\mu_2$					λ_1									
13							μ_2					λ_1								
20					$2\mu_1$				λ_2				λ_1							
21						$2\mu_1$			μ_2					λ_1						
22							μ_1			μ_2					λ_1					
23											$3\mu_2$					λ_1				
30									$3\mu_1$											
31										$2\mu_1$			μ_2							
32											μ_1			$2\mu_2$						
33															$3\mu_2$					

- (1) n, m represent Class II/Class I subscripts
- (2) λ_1, μ_1 are Class II parameters
 λ_2, μ_2 are Class I parameters
- (3) Each main diagonal element is formed as follows:
(Diagonal element) = (Sum of corresponding
row entries)

large nodal models, conventional Gaussian matrix solution techniques become prohibitive.

The algorithm formulated by this procedure is computationally tractable. Closely related to Wongs, et al work [96], a matrix operator B is defined and steady-state data queue length distribution obtained in terms of its eigenvalues and eigenvectors. The P_0 probability vector is determined by Gaussian solution of a $C+1$ system of equations. All other probability vectors are iteratively expressed in terms of P_0 , and the resulting probabilities normed. Measures of effectiveness such as expected number of data/voice transactions in the system or queue, probability of all channels occupied by voice, etc., are then derived using the probability distribution. The algorithm is implemented by a FORTRAN computer program that requires at most $(2C+2)*(2C+2)$ matrices for its calculations.

5.1 Solution Process

The solution process decomposes the rate matrix Γ into several submatrices (Fig. 9). Each submatrix is $(C+1)*(C+1)$ and defined as follows:

$$\Lambda = \begin{bmatrix} \lambda_1 & \bigcirc \\ & \cdot \\ & \cdot \\ \bigcirc & 0 \end{bmatrix}$$

$$A_D =$$

for $n = 0, 1, \dots, C-1$

$A_C =$

$$A_C' = A_C \text{ except last diagonal entry is } -C_{u_2}$$

Four major steps, each requiring development and verification of a computer program, are necessary to complete the solution process. These steps are:

- (a) Algorithm Development
- (b) P_0 Vector Determination
- (c) Steady-State P Vector Determination
- (d) Determination of Effectiveness Measures

5.1.1 Algorithm Development

By multiplying the rate matrix with P vectors, the following equations result:

$$P_0 A_0 + P_1 M_1 = 0 \quad (5.1)$$

$$P_{n-1} \Lambda + P_n A_n + P_{n+1} M_{n+1} = 0, \quad n=1, \dots, C-1 \quad (5.2)$$

$$P_{n-1} \Lambda + P_n A_C + P_{n+1} M_C = 0, \quad n=C, C+1, \dots, N-2 \quad (5.3)$$

$$P_{n-2} \Lambda + P_{n-1} A_C + P_n \hat{M} = 0, \quad n=N-1 \quad (5.4)$$

$$P_0 Q + P_{n-1} \Lambda + P_n A_C = 0, \quad n=N \quad (5.5)$$

Equation 5.3 can be rewritten as:

$$P_{n+1} = P_n H_1 + P_{n-1} H_2, \quad n=C, C+1, \dots, N-2 \quad (5.6)$$

$$\text{where } H_1 = -A_C M_C^{-1} \text{ and } H_2 = -\Lambda M_C^{-1}$$

Let $X_n = (P_n, P_{n+1})$, $n=0, 1, 2, \dots$

Define a matrix operator B $(2C+2) \times (2C+2)$ partitioned as

$$B = \begin{bmatrix} 0 & H_2 \\ I & H_1 \end{bmatrix}$$

Then from 5.6, $X_{n+1} = X_n B$, for $n=C-1, C, \dots, N-2$

$$\text{Now } X_C = X_{C-1} B \quad (5.7)$$

$$\begin{aligned} X_{C+1} &= X_{C-1} B^2 \\ &\vdots \\ X_{C+k} &= X_{C-1} B^{k+1}, \text{ for } k=0, 1, 2, \dots \\ &\vdots \\ X_{N-1} &= X_{C-1} B^{N-C} \end{aligned}$$

$$\text{Let } G_n = \begin{bmatrix} 0 & -\Lambda M_{n+1}^{-1} \\ I & -A_n M_{n+1}^{-1} \end{bmatrix} \text{ be a } (2C+2) \times (2C+2) \text{ operator,}$$

where $n=0, 1, \dots, C-2$

Rewriting equation 5.2 results in:

$$\begin{aligned} P_{n+1} &= -P_n A_n M_{n+1}^{-1} - P_{n-1} \Lambda M_{n+1}^{-1} \\ \implies (P_{n+1}, P_{n+2}) &= (P_n, P_{n+1}) G_{n+1} \end{aligned}$$

$$\text{Then } X_1 = X_0 G_1$$

$$X_2 = X_1 G_2 = X_0 G_1 G_2$$

$$X_{C-1} = X_0 G_1 G_2 \dots G_{C-1}$$

Using 5.7 in combination with the preceeding results in

$$\begin{aligned}
 x_C &= x_0 G_1 \dots G_{C-1} B \\
 x_{C+1} &= x_0 G_1 \dots G_{C-1} B^2 \\
 &\vdots \\
 x_{N-2} &= x_0 G_1 \dots G_{C-1} B^{N-C} \\
 \therefore x_{N-1} &= x_0 G_1 \dots G_{C-1} B^{N-C} B'
 \end{aligned}$$

where B' is formed similarly to B except M' replaces M_C in the calculation of H_1 and H_2 .

Now from 5.1:

$$\begin{aligned}
 x_0 &= (p_0, p_1) = (p_0, -p_0 A_0 M_1^{-1}) = p_0 (I, -A_0 M_1^{-1}) \\
 \therefore x_{N-1} &= p_0 (I, -A_0 M_1^{-1}) G_1 \dots G_{C-1} B^{N-C} B'. \quad (5.8)
 \end{aligned}$$

Because equation 5.5 is redundant delete it.

Thus $p_{N-1} A + p_N A_C = 0$

$$\begin{aligned}
 \Rightarrow (p_{N-1}, p_N) \begin{bmatrix} A \\ A_C \end{bmatrix} &= 0 \\
 \Rightarrow x_{N-1} \begin{bmatrix} A \\ A_C \end{bmatrix} &= 0 \quad (5.9)
 \end{aligned}$$

Now combining 5.8 and 5.9 results in

$$P_0(I, -A_0 M_1^{-1}) G_1 \dots G_{C-1} B^{N-C} B^T \begin{bmatrix} A \\ C \end{bmatrix} = 0 \quad (5.10)$$

In essence, the solution to equations 5.1 - 5.5 in terms of 5.10 comprise the algorithm development.

5.1.2 P_0 Vector Determination

The P_0 Vector is determined by a computer program that implements equation 5.10. The validity of the computer program was checked by implementing a small system ($C=3$) and verifying the solution by another computer program that solved the system using Gaussian elimination.

5.1.3 Steady-State P Vector Determination

The determination of P vectors P_0, P_1, \dots, P_N requires the development of an efficient technique to calculate matrix B^{N-C} . The technique utilized is based on Wong's work [96] that determines B^N through a similarity transformation $B = RJR^{-1}$, where R and R^{-1} are non-singular matrices and J a matrix of distinct eigenvalues.

Lemma: The characteristic matrix of B , $B(z)$, has the same eigenvalues as those of a reduced z -matrix $D(z)$ [96].

Proof: The characteristic matrix for B is

$$B(z) = (zI - B) = \begin{bmatrix} zI & -H_2 \\ -I & zI - H_1 \end{bmatrix}$$

$$H_2^- \quad \begin{array}{c} -\lambda_1 \\ C_{11} \quad -\lambda_1 \\ (C-1)_{11} \end{array} \quad \begin{array}{c} \text{---} \\ \text{---} \\ 0 \end{array}$$

From 5.11

$$H(z) = \frac{z^2 - \frac{z(\lambda_1 + C_{11})}{C_{11}} + \frac{\lambda_1}{C_{11}}}{z^2 - \frac{z(\lambda_1 + \mu_2 + (C-1)\mu_2)}{(C-1)\mu_1} + \frac{\lambda_1}{(C-1)\mu_1}} + \frac{2\mu_2 z}{(C-1)\mu_1}$$

$$\begin{array}{c} \frac{2\mu_2 z}{(C-1)\mu_1} \quad \frac{z^2 - \frac{z(\lambda_1 + (C-1)\mu_2 + \mu_1)}{\mu_1} + \frac{\lambda_1}{\mu_1}}{z^2 - \frac{z\lambda_1}{C_{11}} + \frac{\lambda_1}{C_{11} + 1}} \end{array}$$

The eigenvalues are obtained by solving each quadratic equation. By inspection, the eigenvalues in the last diagonal entry are $z=0$ and $z=-\lambda_1/(C_{n2}+\lambda_1)$. The eigenvalues in the k th position on the diagonal are determined by solving the general equation:

$$(C-k)\lambda_1 z^2 - z(\lambda_1 + k\mu_2 + (C-k)\mu_1) + \lambda_1 = 0 \quad (5.12)$$

These eigenvalues constitute the diagonal elements of matrix J .

For notational convenience, let $S = P^{-1}$. Then $B = RJS$ and the k th column of R is the k th eigenvector [96]. Post multiplying each side by P results in $BR = RJ$ decomposed as

$$\Rightarrow \begin{bmatrix} 0 & H_2 \\ I & H_1 \end{bmatrix} \begin{bmatrix} R_1 & R_2 \\ R_3 & R_4 \end{bmatrix} = \begin{bmatrix} R_1 & R_2 \\ R_3 & R_4 \end{bmatrix} \begin{bmatrix} J_0 & 0 \\ 0 & J_1 \end{bmatrix}$$

where $R_i = (R_{i,1}, R_{i,2}, \dots, R_{i,C+1})$, $i=1, 2, 3, 4$

The following equations can then be derived.

$$H_2 R_3 = R_1 J_0 \quad (5.13)$$

$$R_1 + H_1 R_3 = R_3 J_0 \quad (5.14)$$

$$H_2 R_4 = R_2 J_1 \quad (5.15)$$

$$R_2 + H_1 R_4 = R_4 J_1 \quad (5.16)$$

Post multiplying 5.14 by J_0 and substituting from 5.13

AD-A107 254

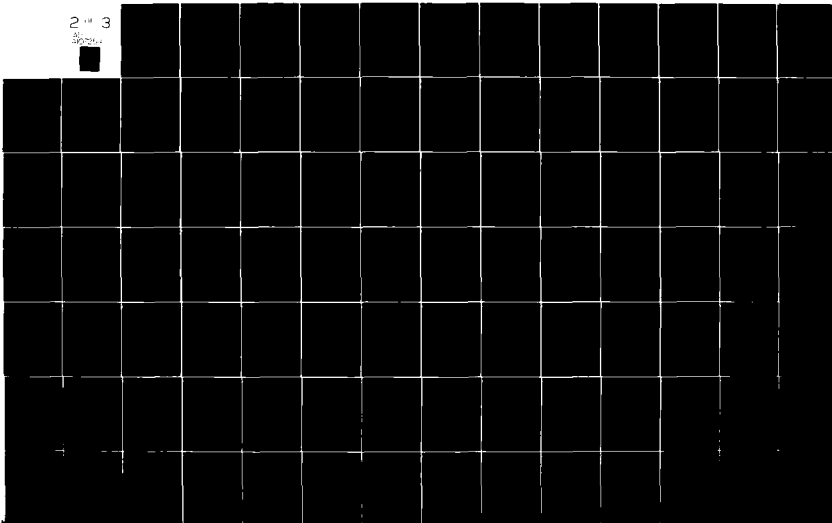
AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH
ANALYSIS OF FLOW BEHAVIOR WITHIN AN INTEGRATED COMPUTER-COMMUNI--ETC(U)
MAY 79 C A CLABAUGH
AFIT-CI-79-2130

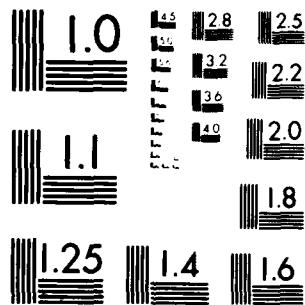
F/G 17/2

UNCLASSIFIED

NL

2 3





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

results in:

$$R_1 J_0 + H_1 R_3 J_0 = R_3 J_0^2$$

$$R_3 J_0^2 - H_1 R_3 J_0 - H_2 R_3 = 0$$

Now since J is in reality eigenvalues z , and by looking at the k th column

$$(z_k^2 I - z_k H_1 - H_2) R_{3,k} = 0$$

$$\text{or } D(z_k) R_{3,k} = 0$$

For notational simplicity represent $D(z_k)$ as

$$D(z_k) = \begin{bmatrix} d_1 & & & & & \\ d_1' & d_2 & & & & \\ & d_2' & . & & & \\ & & & . & & \\ & & & & . & \\ & & & & & d_C' & d_{C+1} \end{bmatrix}$$

where d, d' equate to previously defined $D(z)$ elements,

Now eigenvector $R_{3,k} =$

$$\begin{bmatrix} r_{1,k} \\ r_{2,k} \\ . \\ . \\ . \\ r_{C+1,k} \end{bmatrix}$$

where z_k varies over eigenvalues in J_0 .

Then $r_{1,k} = r_{2,k} = \dots = r_{\ell-1,k} = 0$

where ℓ is the index of the row whose eigenvalue gave r_k .

Let $r_{\ell,k} = 1$ and $r_{n,k} = -d_n r_{n-1} / d_n$ for all $n > \ell$.

Thus eigenvector R_3 elements can be determined, and likewise from 5.16

$$R_2 J_1 + H_1 R_4 J_1 = R_4 J_1^2 \implies (z_k^2 I - z_k H_1 - H_2) R_{4,k} = 0.$$

The same iteration scheme (varying z_k over J_1 eigenvalues) gives eigenvector R_4 .

After building R_3 and R_4 , equations 5.14 and 5.16 are used to build R_1 and R_2 .

To avoid calculating S^{-1} , matrix S is calculated iteratively similar to R . Pre-multiplying each side of $B = RJS$ by S results in $SB = JS$ decomposed as

$$\begin{bmatrix} S_1 & S_2 \\ S_3 & S_4 \end{bmatrix} \begin{bmatrix} 0 & H_2 \\ I & H_1 \end{bmatrix} = \begin{bmatrix} J_0 & 0 \\ 0 & J_1 \end{bmatrix} \begin{bmatrix} S_1 & S_2 \\ S_3 & S_4 \end{bmatrix}$$

where $S_i = (S_{i,1}, S_{i,2}, \dots, S_{i,C+1})$, $i=1, 2, 3, 4$.

The following equations can then be derived

$$S_2 = J_0 S_1 \quad (5.17)$$

$$S_1 H_2 + S_2 H_1 = J_0 S_2 \quad (5.18)$$

$$S_4 = J_1 S_4 \quad (5.19)$$

$$S_3 H_2 + S_4 H_1 = J_1 S_4 \quad (5.20)$$

Substituting 5.17 and 5.18 results in

$$S_{1,k}(z_k^2 - z_k H_1 - H_2) = 0 \implies S_{1,k} D(z_k) = 0.$$

Using an iterative procedure similar to that employed to build R_3 and R_4 , row elements of S_1 and S_3 are determined. Equations 5.17 and 5.19 are used to calculate S_2 and S_4 elements. Since each row of S represents some multiplicity of each corresponding eigenvalue, it is necessary to compute $I=RS$ and divide S by the resultant diagonal elements of I to obtain a true S . Upon completing this portion of the software, all probability vectors needed for steady-state representation are obtained using equations 5.1 - 5.5. These probability vectors are then normed forming a probability distribution.

5.2 Determination of Effectiveness Measures

The following steady-state effectiveness measures are considered relevant:

- (1) Number of data transactions in the system
- (2) Number of data transactions in the queue
- (3) Number of voice calls in the system
- (4) Probability of all channels busy with voice calls
- (5) Expectation of data transaction in the system
given all channels busy with voice

(6) Probability of all channels almost full with voice calls

(7) Expectation of data transactions in the system given all channels almost full with voice calls.

These measures are obtained from the steady-state probability distribution for the system [32]:

$$L_{\text{Data}} = \sum_{n=0}^N n \sum_{m=0}^C P_{n,m} \quad (5.21)$$

$$L_{\text{Data}}^q = \sum_{m=0}^C \sum_{n=C-m}^N [n-(C-m)] P_{n,m} \quad (5.22)$$

$$L_{\text{Voice}} = \sum_{m=0}^C m \sum_{n=0}^N P_{n,m} \quad (5.23)$$

$$P\{m=C\} = \sum_{n=0}^N P_{n,C} \quad (5.24)$$

$$E[n|m=C] = \frac{\sum_{n=0}^N n P_{n,C}}{\sum_{n=0}^N P_{n,C}} \quad (5.25)$$

$$P\{m \geq C-2\} = \sum_{n=0}^N P_{n,C-2} + P_{n,C-1} + P_{n,C} \quad (5.26)$$

$$E[n|m \geq C-2] = \frac{\sum_{n=0}^N n [P_{n,C-2} + P_{n,C-1} + P_{n,C}]}{\sum_{n=0}^N P_{n,C-2} + P_{n,C-1} + P_{n,C}} \quad (5.27)$$

5.3 Conclusions

The development of computationally tractable computer-communication network mathematical models is difficult due to numerous network variables and assumptions [36]. Used

in conjunction with the network simulator, computer implementation of the model developed herein provides useful steady-state effectiveness measures, which can be used to confirm simulation output or provide bounds on simulation expectations. It is implemented by a highly portable FORTRAN computer program.

CHAPTER VI

DEVELOPMENT OF AN INTEGRATED FLOW CONTROL SCHEME

6. The Need For a Flow Control Scheme

The statistical analysis (Figs. 15-20 and Appendix E tables) indicates that controls are necessary to manage nodal activity at the packet nodes. With no flow controls other than those generated by varying system parameters, Fig. 15 shows that subscriber packet delays become significant. Although the model does not test for abnormal network situations such as path failures, subscriber malfunctions, etc., they clearly contribute further to the problem. Analogous to a store-and-forward network, controls over packet nodal storage and subscriber activity in an integrated system are required. Independent of any control intelligence within the subscriber environment or circuit switch blocking factors, packet switch loads must be regulated.

6.1 Utility of Existing Strategies

The model design is based on an underlying circuit-switching structure. All flow control layers are maintained; however, the subnet-subnet and subnet-subscriber layers are utilized only during the routing process or abnormal network conditions. This situation exists because subnet nodes permit no data queues and exercise limited network control.

The lack of sophistication in these layers eliminates as potential solution techniques all subnet controlling strategies except nodal limits and nodal buffer monopoly. These strategies are node controlling and not dependent on nodal coordination or interplay. Within the subscriber-subscriber flow layer, all subscriber controlling strategies but the window scheme are eliminated as potential solution techniques. Only the window strategy possesses the flexibility for controlling link bandwidth and subscriber-subscriber communications.

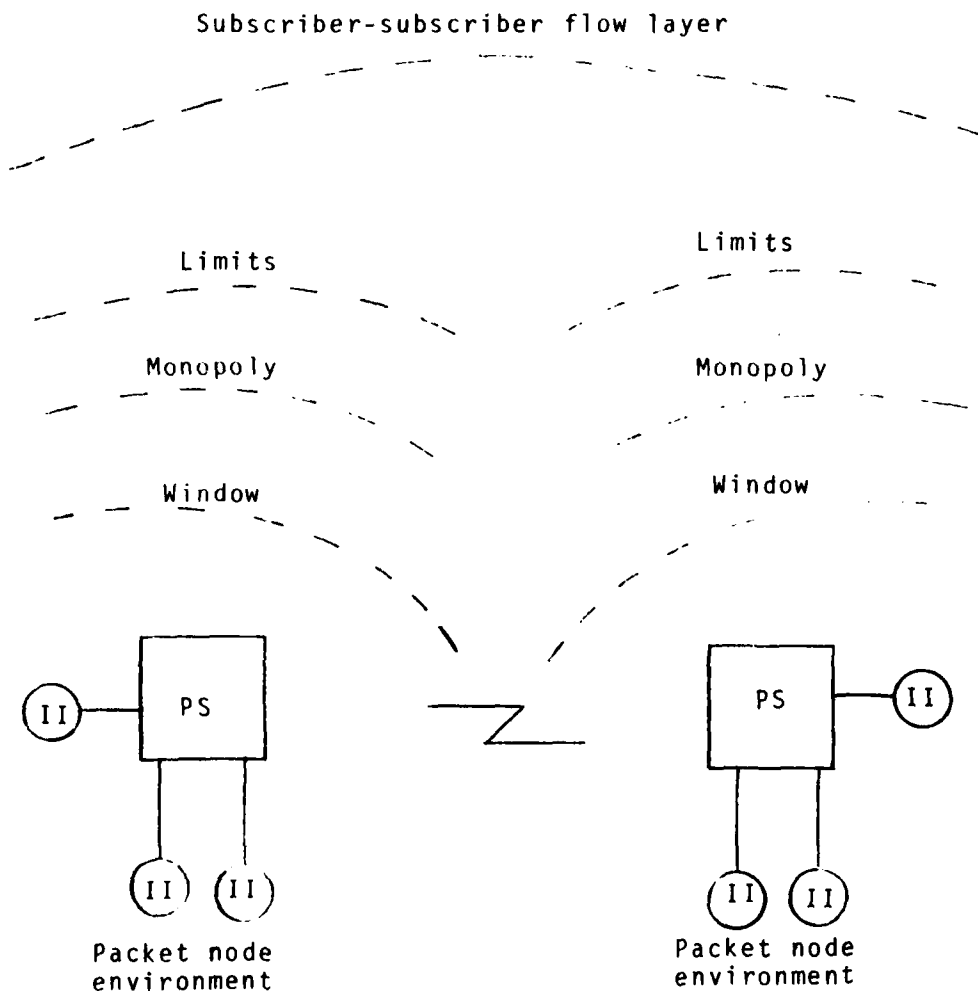
6.2 A Ring Control Scheme Proposal

Packets are assumed to contain only routing and/or subscriber coordination information; thus, a network flow control scheme based on packet composition is not considered.

The subscriber-subscriber flow control layer is directly involved with subscriber process multiplexing/demultiplexing and storage availability. The framework for an integrated flow control scheme consists of requirement-capability matching between this layer and nodal limit, nodal buffer monopoly, and window strategies. A flow control hierarchy can be constructed consisting of the following strategy rings (Fig. 10):

- (a) Nodal Limits Ring
- (b) Nodal Monopoly Ring
- (c) Window Ring

Fig. 10. Flow control ring scheme



6.2.1 Nodal Limits Ring

Foremost is the requirement that a node not become saturated. The nodal limits strategy ring insures the node will not become overloaded. This is accomplished by rejecting input packets when a nodal threshold is exceeded. The rejection process consists of discarding all packets received above a prescribed nodal limit knowing that the window strategy ring will subsequently detect an out-of-order packet condition and effect subscriber controls.

6.2.2 Nodal Monopoly Ring

This control ring monitors all subscriber-link flow connections using a buffer monopoly matrix whose rows and columns represent subscriber and link loadings respectively. With a monopoly matrix three flow rates can be monitored:

- (1) Subscriber Buffer Load
- (2) Channel loading
- (3) Subscriber-Channel Dominance

Buffer loading by each subscriber terminated to a packet node can be measured by summing across a matrix row. This reflects the total buffer load by a single subscriber on several slots.

Each matrix column represents the packet waiting load for a given channel. If the channel loading exceeds a prescribed level, the channel can be either upgraded by establishing a new connection using more slots, or the load reduced by discarding packets.

6.2.3 Window Ring

The inner strategy ring exists within the subscriber environment and requires an intelligent mechanism for subscriber-subscriber operational control. This device must be capable of executing the window strategy algorithm for multiple connections. Paramount for strategy ring interplay is the assumption of four well defined window properties [2, 5, 29]:

- (1) Source/destination subscribers efficiently utilize bandwidth by "pushing" or "slowing-down" each other.
- (2) In coordination with nodal strategy rings, any discarded packets are detected by the source and retransmission controls effected.
- (3) Source/destination subscribers can initiate transactions without prior buffer commitment
- (4) Physical end-to-end controls become practical.

6.3 Ring Scheme Implementation

Implementation of the ring scheme hierarchy can be accomplished by instituting nodal and subscriber software controls, and the aforementioned window properties. Software controls consist of:

- (a) Nodal Limit Variables
- (b) Global Monopoly Matrix

6.3.1 Nodal Limit Variables

Nodal limits for each packet node can be determined and implemented by two variables at each node, CURLOD and LODLIM. When the current packet load exceeds the preset value in LODLIM, all incoming packets are discarded until CURLOD is reduced to an acceptable level. Property two of the window ring strategy will detect this condition and effect transmission controls.

6.3.2 Global Monopoly Matrix

The nodal monopoly ring can be implemented through use of a global matrix which relates Class II subscribers (rows) and channels (columns) (Fig. 11). Nodal termination of four subscriber types, (video, bulk, message, and interactive) can be simulated by varying time slot allocations for each type.

6.4 Subscriber Window Development

The window strategy ring is a combination of software/hardware techniques implemented within the subscriber environment. As such, it does not directly impact the model, but its structural capabilities must be described. The control mechanism must be an intelligent device consisting of typical processor elements (CPU, Arithmetic Logic Unit, memory, etc.), and two significant Input/Output (I/O) capabilities:

- (a) Multiple Interrupt Level Capability

Fig. 11. Flow control monopoly matrix

		Channels				
		1	2	3	.	Param(2)
Node 1	Video					
	Bulk					
	Message					
	Packet					
Node 2	Video					
	Bulk					
	Message					
	Packet					
	.					
	.					
	.					
	Param(1)					

(b) Variable I/O Data Rate Terminations

6.4.1 Multiple Interrupt Level Capability

Multiple subscriber connections can be expected. The controller must be able to transfer control to several types and levels of I/O interrupt handlers (video to facsimile, message to facsimile, subscriber transaction x to subscriber data conversion routine y, etc.).

6.4.2 Variable I/O Data Rate Terminations

To reduce the frequency of interrupts associated with terminations of multiple high speed connections, a Direct Memory Access (DMA) capability must exist. This insures that interrupts are properly serviced in an orderly manner, guards against data overrun, and provides the facility for terminating variable speed connections with minimal processor overhead.

6.5 Subscriber Software Development

The software requirements for window strategy implementation consist of those system modules necessary to implement the window strategy algorithm, establish coordination with the host node, and user modules that pertain to subscriber-subscriber communication.

CHAPTER VII

ANALYSIS OF NETWORK BEHAVIOR

7. Network Configuration

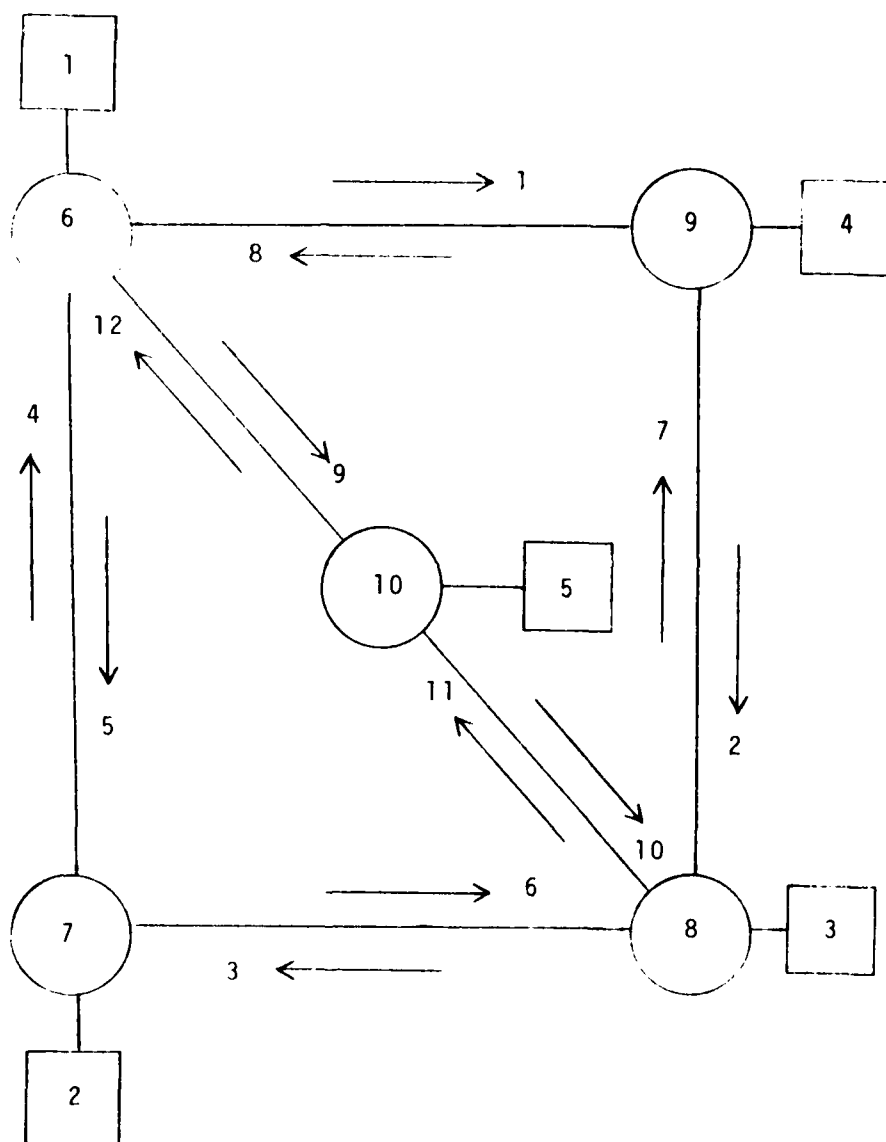
The simulator was designed to accommodate network configurations with variable node and channel arrangements. Because of the numerous system parameters, the simulator can generate empirical data involving multiple flow relationships between arrival patterns, nodal configurations, integration methodology, Voice Digitization Rates, and data/voice time slot ratios. The 10-node network (Fig. 12) configured for this research addresses these complex issues. It provides substantial channel capacity for in-depth flow analysis without having to resort to a dedicated computer facility and corresponding cost. The specific purpose of this configuration was to demonstrate the practicality of an integrated network based on an underlying circuit switch subnet.

7.1 Examination of the Simulator

The following user input parameters and specifications define the baseline performance criteria for a given simulation run:

- (1) Number of Nodes
- (2) Number of Links
- (3) Number of Time Slots

Fig. 12. 10-Node network configuration



- (4) Slot Time
- (5) Cross-Office Signalling Delay
- (6) Voice and Data Arrival Rates
- (7) Simulation Start and End Time
- (8) Steady-State Criteria
- (9) Voice Digitization Rate
- (10) Allocation of Data Transaction Storage
- (11) Service Rates for Voice and Data
- (12) Number of Packets Per Message

In addition, each physically different network configuration must be coordinated with corresponding channel and queue storage requirements. The simulator requires user specification of each network configuration through input table initialization to DESTAB, DSTALT, NODCHL, and SEEDTB. DESTAB and DSTALT tables contain the user defined primary and alternate routing paths for each source/destination pair. The order of node/channel connectivity is placed in NODCHL. SEEDTB contains user specified seeds for each node. These seeds are used by the random number generator. A description of the input table initialization entries is contained in Figs. 13 and 14.

7.1.2 Confidence in the Simulator

The Poisson and geometric generators used in the simulator are based on the combined research of Jackson and Kleinrock for assuming Poisson arrivals and exponential service, and geometric distribution of data transactions for

Fig. 13. 10-Node routing table initialization

		DESTAB									
		DEST →									
		1	2	3	4	5	6	7	8	9	10
Node	1	0	6	6	6	6	0	0	0	0	0
	2	7	0	7	7	7	0	0	0	0	0
	3	8	8	0	8	8	0	0	0	0	0
	4	9	9	9	0	9	0	0	0	0	0
	5	10	10	10	10	0	0	0	0	0	0
	6	0	5	9	1	9	0	5	9	1	9
	7	4	0	6	4	4	4	0	6	4	4
	8	7	3	0	7	11	7	3	0	7	11
	9	8	2	2	0	2	8	2	2	0	2
	10	12	12	10	10	0	12	12	10	10	0
		DSTALT									
		DEST →									
		1	2	3	4	5	6	7	8	9	10
Node	1	0	6	6	6	6	0	0	0	0	0
	2	7	0	7	7	7	0	0	0	0	0
	3	8	8	0	8	8	0	0	0	0	0
	4	9	9	9	0	9	0	0	0	0	0
	5	10	10	10	10	0	0	0	0	0	0
	6	0	1	1	9	1	0	1	1	9	1
	7	6	0	4	6	6	6	0	4	6	6
	8	11	11	0	3	3	11	11	0	3	3
	9	2	8	8	0	8	2	8	8	0	8
	10	10	10	12	10	0	10	10	12	10	0

Fig. 14. Input parameters and specifications for 10-node network

Card 1 - 10 12 48 3 10 50 15 20 0 480000
 99000 32000 1800 180 1000 10

Card 2 06413 46427 86799 19565
 17767 05431 35635 99817
 26803 20505 14523 81949
 49329 28573 16213 78317
 15307 08391 00597 32537
 45611 49883 09303 71715
 36147 68607 98083 58401
 64969 08015 79953 08721
 89837 88159 25241 75379
 10851 50949 06571 37143

Card 12 - 0 6 6 6 6 0 0 0 0 0
 7 0 7 7 7 0 0 0 0 0
 8 8 0 8 8 0 0 0 0 0
 9 9 9 0 9 0 0 0 0 0
 10101010100 0 0 0 0
 0 5 9 1 9 0 5 9 1 9
 4 0 6 4 4 4 0 6 4 4
 7 3 0 711 7 3 0 711
 8 2 2 0 2 8 2 2 0 2
 12121010 012121010 0

Card 22 - 0 6 6 6 6 0 0 0 0 0
 7 0 7 7 7 0 0 0 0 0
 8 8 0 8 8 0 0 0 0 0
 9 9 9 0 9 0 0 0 0 0
 10101010 0 0 0 0 0 0
 0 1 1 9 1 0 1 1 9 1
 6 0 4 6 6 6 0 4 6 6
 1111 0 3 31111 0 3 3
 2 8 8 0 8 2 8 8 0 8
 10101210 010101210 0

Card 32 - 9 8 7 6 7 8 9 610 810 6

where Card 1 = PARAM entries

Cards 2-11 = SEEDTB entries

Cards 12-21 = DESTAB entries

Cards 22-31 = DSTALT entries

Card 32 = NODCHL entries

modeling networks [38, 49]. The random number generator is a commonly accepted pseudo-random number generator [59] that generates a uniformly distributed random variable between 0 and 1. The arrival and service process of external customers at each node is assumed independent of the movement of link customers and independent from node to node. Although this is not true because service time is based on the length of the message, the assumption can be assumed reasonable using Kleinrock's independence assumption [49].

Network simulator performance was validated by establishing voice/data confidence intervals on system output transactions [22, 91]. True data/voice means were determined by the use of standard Poisson generators, and user run time specifications. Confidence intervals on data/voice output nodal observations were used to establish that the simulator was performing within a 95% level. Upper and lower confidence bounds were placed around two distinct arrival patterns, assuming a normal distribution, and the student t (since the sample size was less than 30). Table II shows that the simulator is behaving within a 95% data/voice confidence interval.

Additional verification of simulator output was obtained by configuring the simulator to correspond to the analytic model and comparing the output results. Using a common arrival pattern, the number of Class I/II transactions were closely correlated as follows:

Table II. Calculation of Confidence Intervals

Confidence Interval Statistic [91]

$$\bar{x} \pm t_{\text{crit}} \left(\frac{s}{\sqrt{N-1}} \right)$$

where t = sample size s = sample standard deviation N = sample size $\pm = t_{\text{crit}}$ = critical values of t Arrival pattern one calculation

<u>"true" system</u>	<u>observed system</u>
data = 3075	data = 3093.6 $S_1 = 38.05$
voice = 40	voice = 41.4 $S_2 = 9.07$

95% confidence interval for data is 3040 - 3146

95% confidence interval for voice is 28.8 - 54

Arrival pattern two calculation

<u>"true" system</u>	<u>observed system</u>
data = 18000	data = 17571.6 $S_1 = 471.1$
voice = 80	voice = 76.4 $S_2 = 11.3$

95% confidence interval for data is 16916.7 - 18226.4

95% confidence interval for voice is 64.29 - 95.707

Steady-state comparison

	Analytic model	Simulator
Calls in system	15	12.8
Data transactions in system	1	.5

The confidence interval validation augmented by analytical model/simulator correspondence provided sufficient verification of the program logic and computer code.

7.1.3 Description of Simulator Output

Output from the simulator consists of (1) individual channel information relating the number of data and voice transactions over that channel with corresponding utilization, (2) packet node statistics to include packet delays, current and cumulative loading by node and source/destination pairs, and (3) circuit node voice information relating the number in the system, blocking, and total calls. Appendix D contains the output for a typical simulation run.

7.2 Analysis of Network Flow

The sources of relevant parameters and assumptions for the 10-node network were evaluated as follows:

1. The frame interval was 10 milliseconds. This was consistent with several related simulations [10, 18, 36, 60, 83].
2. Packet and circuit switches had common user-to-network call set-up/disconnect procedures with a 50 milliseconds cross-office time and a 32 Kb/sec

CCIS rate. This CCIS rate represents a capability of new generation LSI digital switches [60, 79, 84].

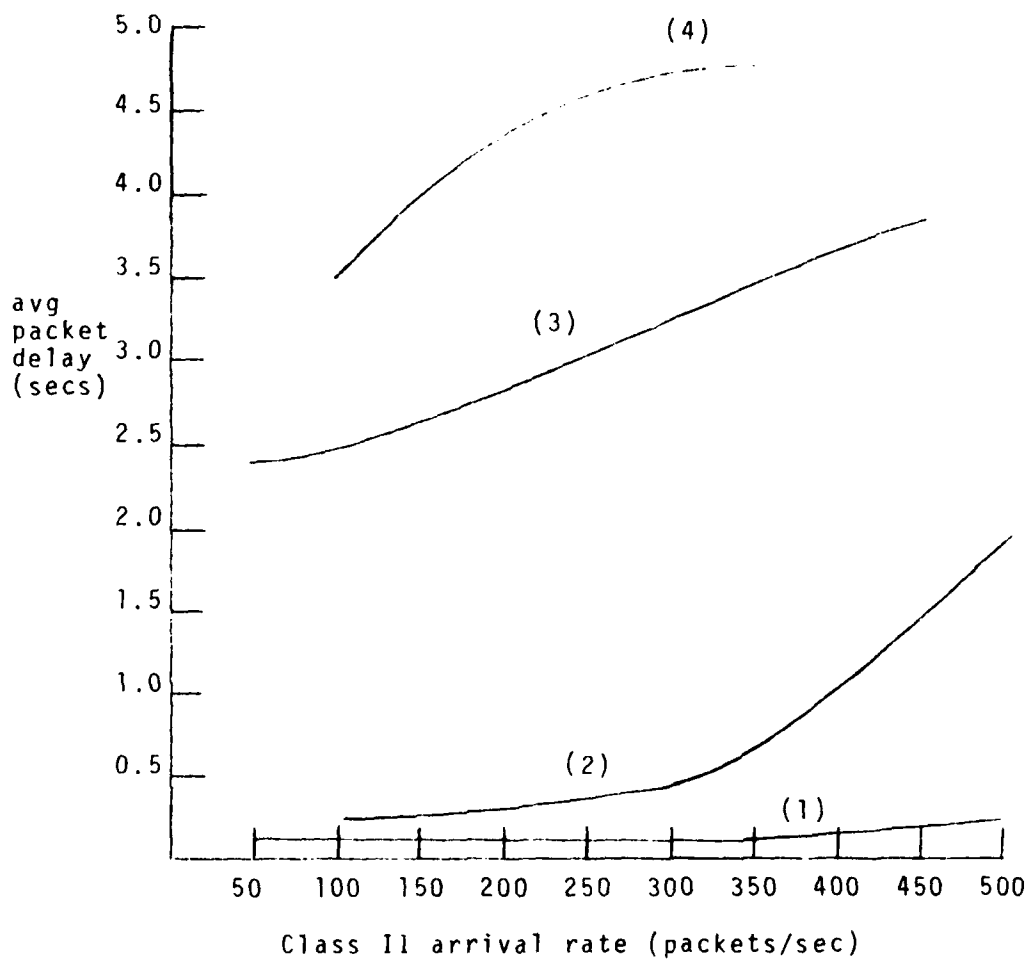
3. A competitive frame management design was implemented because it offers greater potential for efficient use of the T1 link over a wide range of traffic flow [21, 60, 83, 87].

From Figs. 15-20 and tables in Appendix E, the impact on flow behavior using different arrival patterns can be observed. High voice rates (20/min), independent of data flow dominate the channels, resulting in high packet delays, blocking, and channel utilization. Low voice rates (5-15/min) and moderate packet flow (500/sec) result in low blocking, delays, and channel utilization. Within these ranges, a system can be designed for this network that supports moderate voice and data activity. In essence, the basis for grade-of-service (channel utilization, delay, throughput) can be established by network tuning with the simulator.

7.3 Analysis of Analytic Output

The need for analyzing network behavior in computer-communications networks has spurred the development of network simulators because the numerous variables preclude precise analytic solution. Decomposition-synthesis, however, is an analytic approach that has been used to reduce simulation costs.

Fig. 15. Packet delay



Class II arrival rate (packets/sec)

Arrival pattern

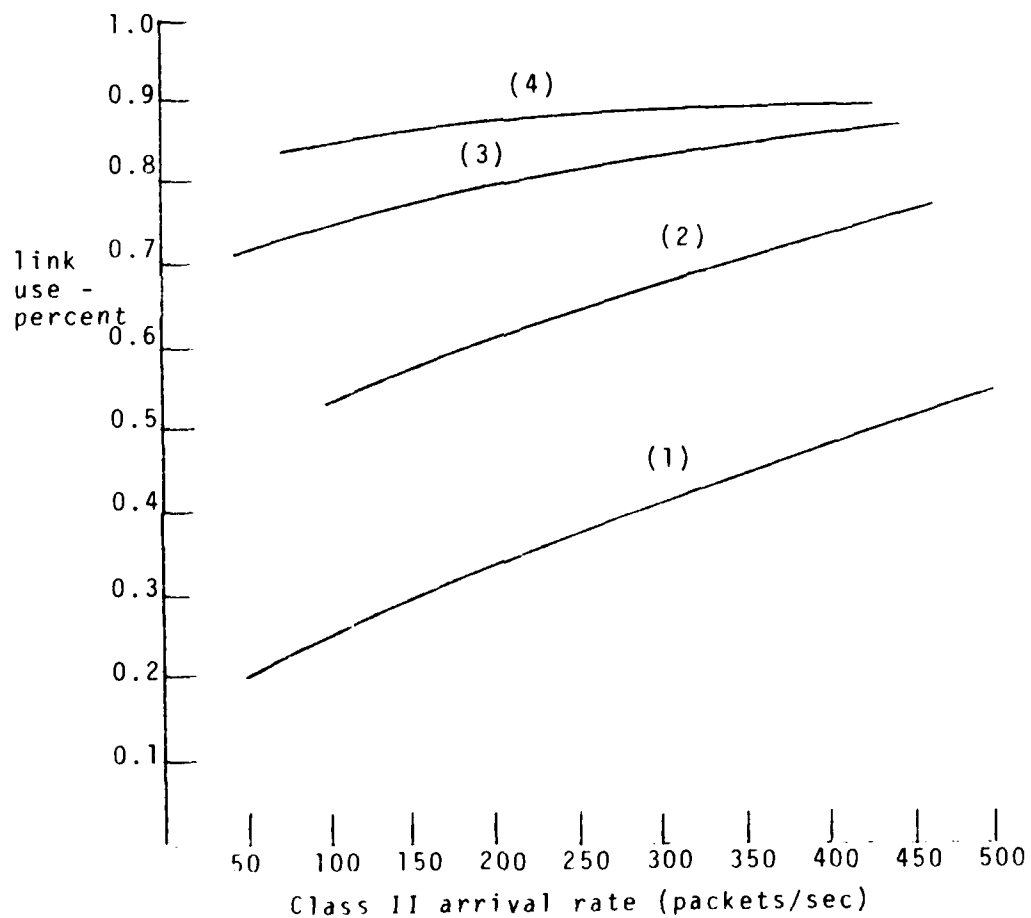
Class I

Class II

- (1) 5/min
- (2) 10/min
- (3) 15/min
- (4) 20/min

as
shown

Fig. 16. Link utilization



Class I

- (1) 5/min
 (2) 10/min
 (3) 15/min
 (4) 20/min

Class II

as
 shown

Fig. 17. Packet throughput

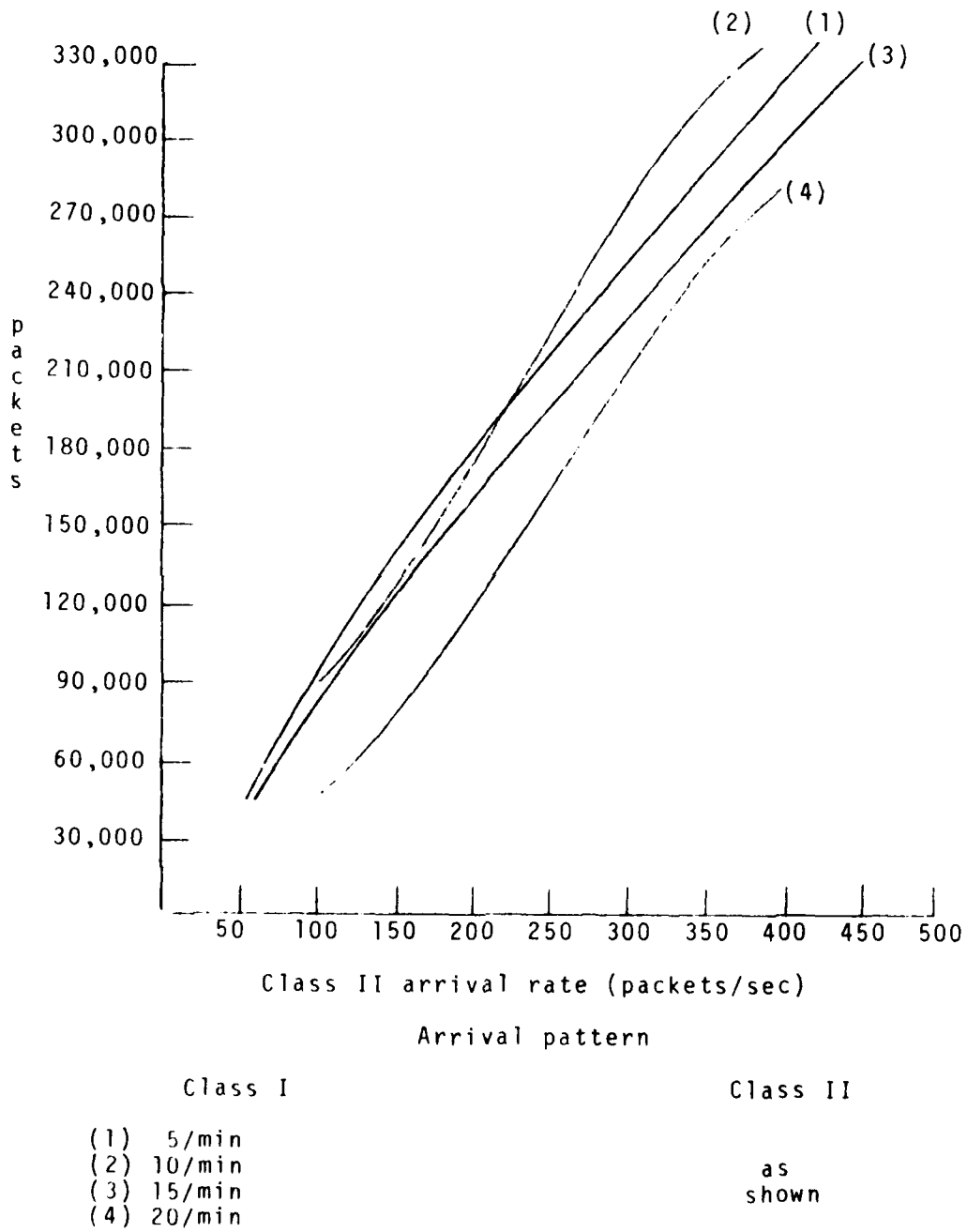
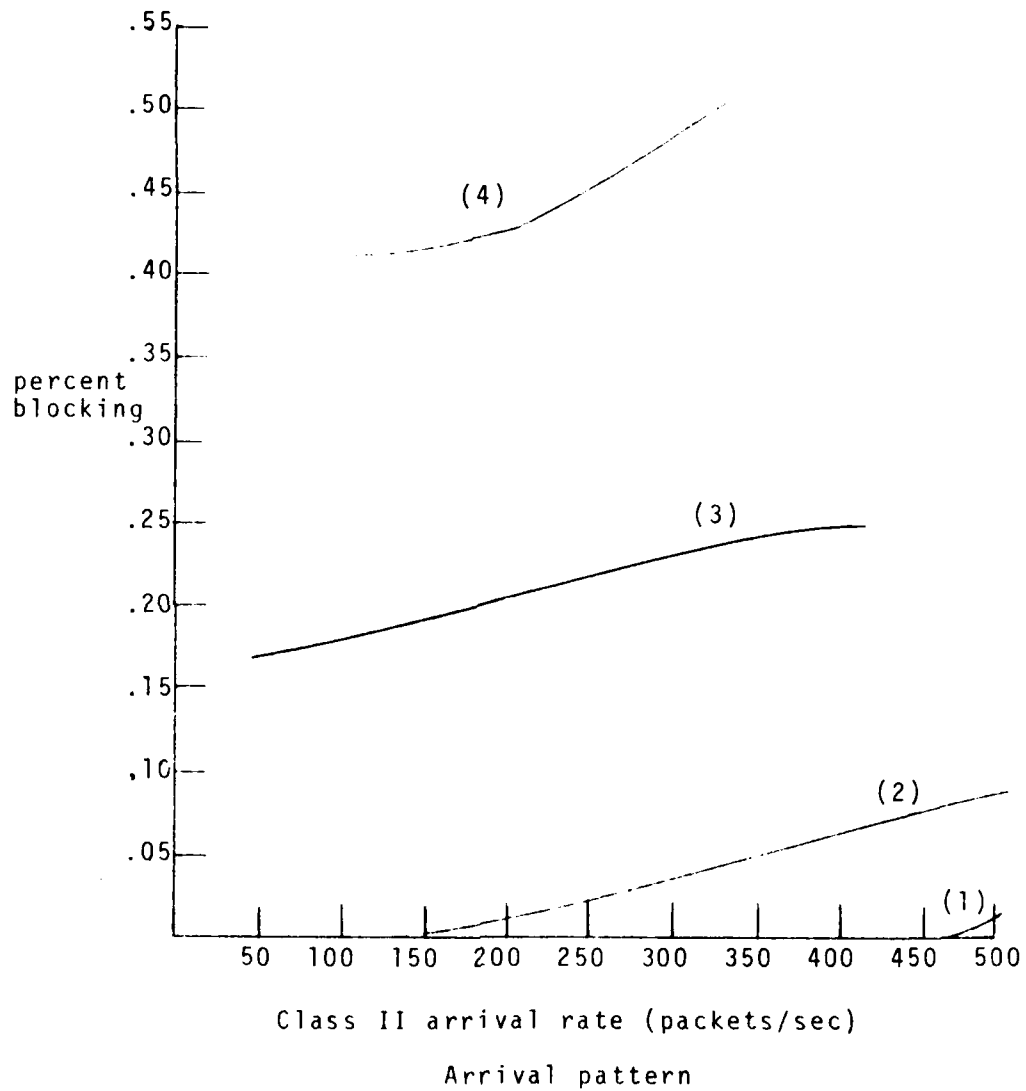


Fig. 18. Class I blocking



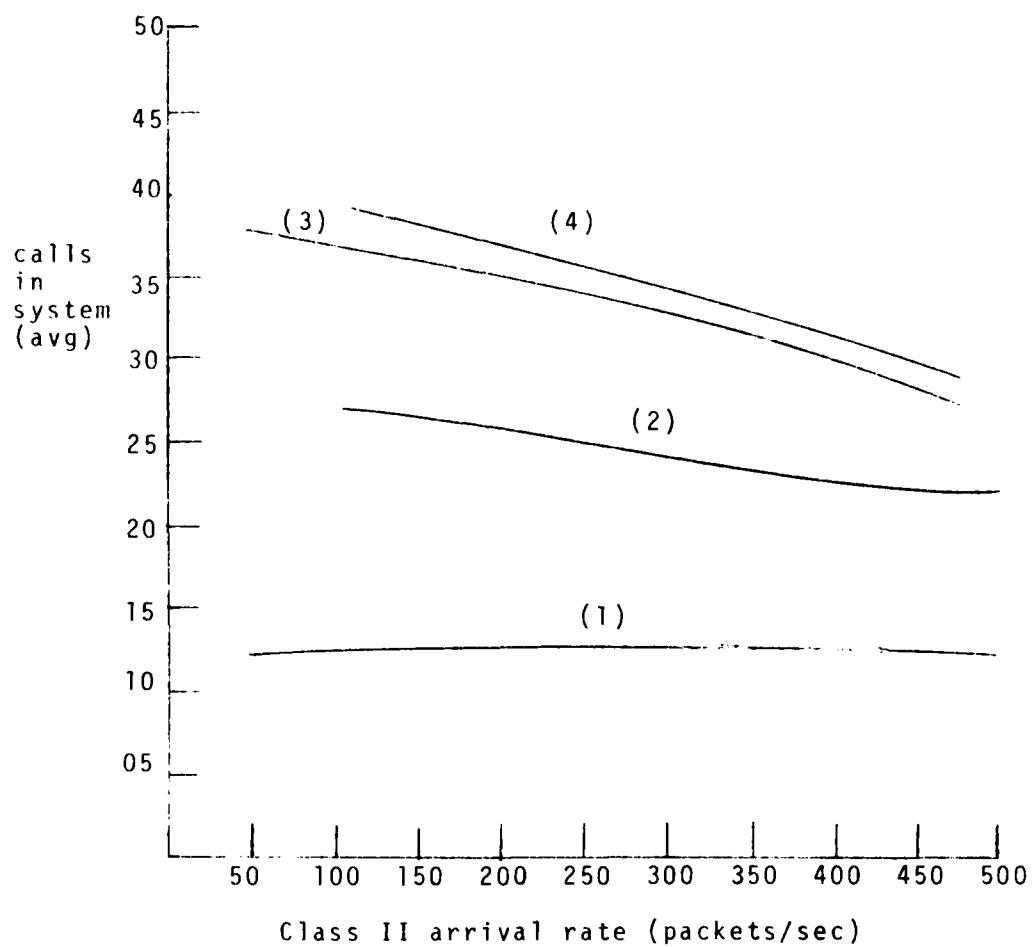
Class I

Class II

- (1) 5/min
- (2) 10/min
- (3) 15/min
- (4) 20/min

as
shown

Fig. 19. Calls in system



Arrival pattern

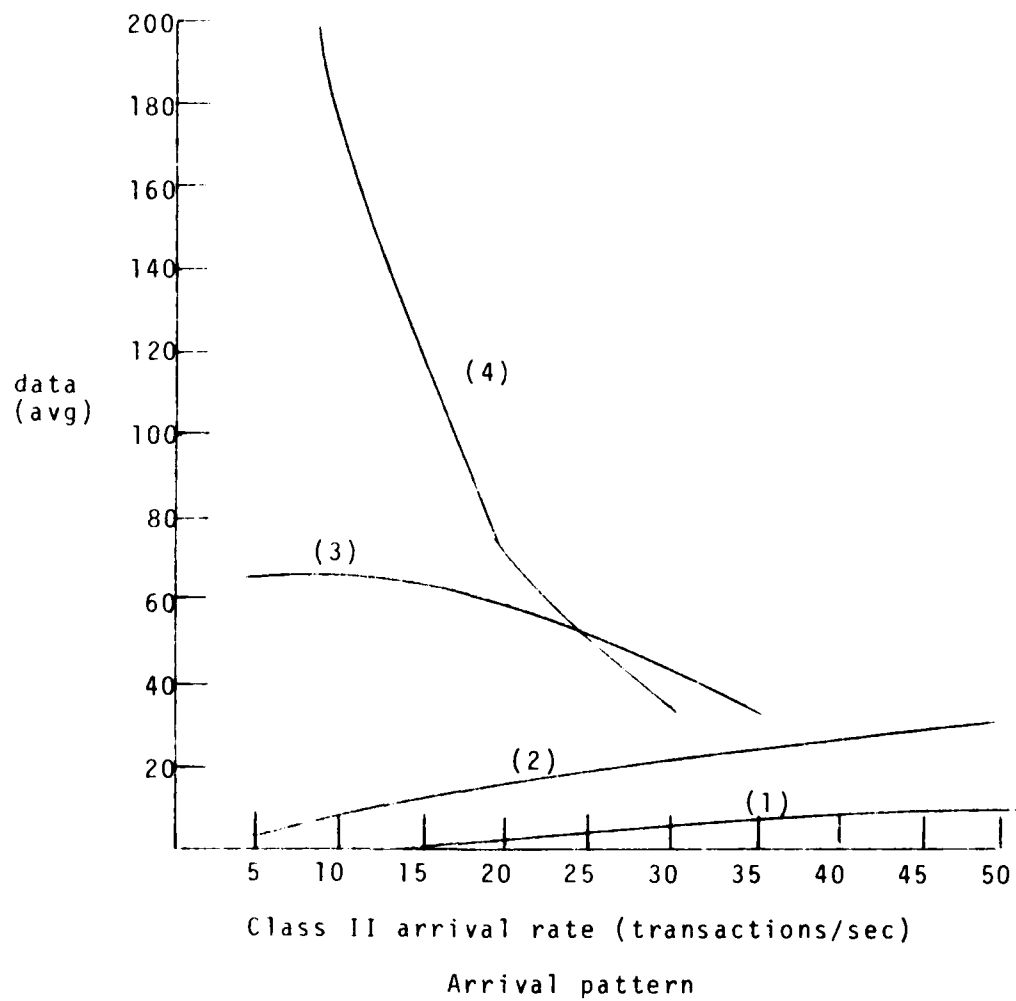
Class I

Class II

- (1) 5/min
- (2) 10/min
- (3) 15/min
- (4) 20/min

as
shown

Fig. 20. Data transactions in system



Class I		Class II	
(1)	5/min	as shown	
(2)	10/min		
(3)	15/min		
(4)	20/min		

7.3.1 Decomposition-Synthesis Approach

Using this approach, a network is decomposed into nodal configurations. The results obtained are synthesized to form a composite solution. Justification for this approach has been provided by Jackson [38], and is based on standard Markovian assumptions of independently exponentially distributed arrival and service times in a system that is assumed to be stable. For the integrated environment this approach was employed. Computer implementation of the analytic model can be used to derive important steady-state measures at each node.

7.3.2 Utility of the Analytic Model

An integrated network based on an underlying circuit switched subnet has not yet been implemented. Analytic tools that have been used to parameterize an integrated network are still in their infancy. The most authoritative work to date has been done by Fischer [21], who treats the network as a queueing problem. He proposes a solution technique involving generating functions, a queueing "gate", and variably integrated frame methodology to analyze network behavior.

The analytic model proposed in this research uses a rate matrix and linear matrix concepts to permit tractable examination of steady-state relationships between voice and data traffic flow. The software implementation of the analytic model is written in FORTRAN and highly portable.

The user specifies arrival and service rates, and channel size from input parameters. The use of a steady-state variable (MLIMIT) allows the user to determine steady-state by noting changes to effectiveness measures before and after MLIMIT adjustments.

As with all analytic network tools, it does have limitations. Firstly, the model is formulated using a one to one time slot correspondence between voice and data. If this correspondence is altered, meaningful output can be obtained by adjusting arrival and service rates to correspond to the new slot relationship. Secondly, double precision accuracy is required for two reasons:

- (1) The iterative calculation of matrices G_1, \dots, G_{C-1} for large channel (C) sizes results in considerable loss of precision.
- (2) The eigenvalues used in the computation of $B=RJS$ are small and close in magnitude.

Finally, software implementation of the model requires four matrices dimensioned $(2C+2) * (2C+2)$ and several $(C+1) * (C+1)$ matrices.

In summary, the need for double precision accuracy and potentially large matrices may demand considerable CPU resources (one megabyte upwards). Appendix C contains steady-state output for various channel sizes.

7.4 Summary

The simulation and analytic models are useful tools for

studying network behavior. Software implementations of these models can be used jointly to characterize traffic behavior in an integrated environment. Analysis of the 10-node configuration indicates that an integrated computer-communications network based on an underlying circuit switch subnet is feasible within a range of traffic flow.

CHAPTER VIII

CONCLUSIONS AND RECOMMENDATIONS

8. Conclusions

The primary objective of this research was to design an integrated network and to analyze the traffic behavior.

Initial investigation was devoted to a review of flow controls in existing store-and-forward and circuit switched networks. As a result of this review, prominent flow control strategies were categorized as subnet or subscriber controlling. This classification provided a framework for potential network regulating techniques in an integrated environment.

Next, it was necessary to develop a simulation model that characterized network behavior. The developed model and its implementation is based on common call management and traffic integration on trunk lines using an underlying circuit switch subnet. It provides substantial flexibility for inter-relating numerous network design features and parameters. The resulting empirical data permits in-depth analysis of several network problem areas.

Another significant goal of this research was the formulation and computer implementation of an analytic tool that can be used to derive system effectiveness measures.

Analysis of summary information from the 10-node model demonstrated that a circuit switched integrated network was practical within a range of data/voice transaction flow. A

need for positive flow controls was also established. Using the flow control classification framework, a ring control hierarchy was proposed. This hierarchy manages controls in the two major areas of concern - nodal overloading and subscriber dominance. Although the hierarchy was not implemented, an implementation scheme was proposed.

3.1 Recommendations

Because of the increased channel capacities and speeds in an integrated network, considerable resources must be expended to collect simulated statistics. This research has shown that dedicated resources are extremely necessary for in-depth modeling and analysis of various network configurations.

The integration methodology used in the model does not incorporate a priority or preemption capability. The design of slot allocations should be researched to include priorities of subscriber traffic and probable system saturation conditions.

A ring flow control hierarchy or related management control mechanism is necessary, and such an implementation is recommended.

The interspersing of sub-classes of data traffic to voice slots needs to be investigated.

With the aid of the network simulator and analytic model, further research into the numerous network design features and parameter inter-relationships needs to be

conducted.

Configuration of the network topology (nodes, links, connectivity, primary/alternate routing paths, etc.) must be manually accomplished. Development of an automated pre-processor is highly recommended to facilitate implementation of various network topologies.

Although not specifically addressed in this dissertation, the following related issues impact network design and require investigation:

1. Voice multiplexing of time slots
2. Development of an optimal routing scheme
3. Design of system timing
4. Nodal packet queue management
5. Nodal design
6. Network flow control

REFERENCES

1. Andrews, Frank B., and Cooper, Chris G. Probing NCR's distributed network architecture. Data Communications, (April 1978), 49-59.
2. Belsnes, Dag. Flow control in the packet switching networks. Communications Networks, (1975), 349-361.
3. Bocker, Peter. Data traffic in communication networks. Proc. of World Telecommunications Forum, Geneva, Switzerland, (1975), 2.3.1.1-2.3.1.8.
4. Branscomb, Lewis M. Trends and developments in computer/telecommunications technologies. Proc. of the OECD Conference, (Feb. 4-6, 1975), 57-77.
5. Cerf, Vinton G., and Kahn, Robert E. A protocol for packet network interconnection. IEEE Trans. on Comm. Com-22, 5 (May 1974), 637-647.
6. Chatterjee, A., Georganas, N.D., and Verma, P.K. Analysis of a packet-switched network with end-to-end congestion control and random routing. IEEE Trans. on Comm. Com-23, 12 (Dec. 1977), 1485-1489.
7. Chu, Wesley W., and Konheim, Alan G. On the analysis and modeling of a class of computer communication systems. IEEE Trans. on Comm. Com-20, 3 (June 1972), 645-660.
8. Cicchetti, G.B., and Lubarsky, A.R. Hybrid integrated digital network. Proc. of World Telecommunications Forum, Geneva, Switzerland, (1975), 2.3.7.1-2.3.7.5.
9. Closs, Felix. Message delays and trunk utilization in line-switched and message-switched data networks. Proc. of First US-Japan Computer Conference, (1972), 524-529.
10. Coviello, Gino J., and Vena, Peter A. Integration of circuit/packet switching by a SENET (Slotted Envelope Network) concept. NTC-75, New Orleans, LA., (Dec. 1-3, 1975), 42-12 - 42-17.
11. Dahlbom, C.A., and Ryan, J.S. History and description of a new signalling system. Bell System Technical Journal 57, 2 (Feb. 1978), 225-250.
12. Davies, Donald Watts. The control of congestion in packet-switching networks. IEEE Trans. on Comm. Com-20, 3 (June 1972), 546-550.

13. Davies, Donald W., and Barber, Derek L. Communication Networks for Computers, John Wiley and Sons, London, England, (1973), 382-421.
14. Doll, Dixon R. Data Communications - Facilities, Networks, and Systems Design, John Wiley and Sons, New York, NY., (1978), 255-304.
15. Doll, Dixon R. Multiplexing and concentration. Proc. of IEEE 60, 11 (Nov. 1972), 1313-1321.
16. Elovitz, Honey S., and Heitmeyer, Constance L. What is a computer network? NTC-74, San Diego, CA., (Dec. 2-4, 1974), 1007-1014.
17. Emshoff, James R., and Sisson, Roger L. Design and Use of Computer Simulation Models, The Macmillan Company, New York, NY., (1970), 5-20.
18. Esterling, R., and Hahn, P. A comparison of digital data network switching alternatives. NTC-75, New Orleans, LA., (Dec. 1-3, 1975), 42-8 - 42-11.
19. Fick, H., Munchen, F.R., and Siemens, A.G. Structures and operating principles of networks for data traffic. ICCC-74, Stockholm, Sweden, (Aug 12-14, 1974), 525-533.
20. Fischer, M.J., and Harris, T.C. A model for evaluating the performance of an integrated circuit-and packet-switched multiplex structure. IEEE Trans. on Comm. Com-24, 2 (Feb. 1976), 195-202.
21. Fischer, M.J., and Harris, T.C. An analysis of an integrated circuit-and packet-switched telecommunications system. NTC-75, New Orleans, LA., (Dec. 1-3, 1975), 42-18 - 42-23.
22. Fishman, George S. Concepts and Methods - Discrete Event Digital Simulation, John Wiley and Sons, New York, NY., 1973.
23. Frank, Howard. Plan today for tomorrows data/voice nets. Data Communications 7, 9 (Sep. 1978), 51-62.
24. Frank, Howard, and Gitman, Israel. Integrated DoD voice & data networks, Rept. AD-A039329, NTIS, (March 1977), 1.1-1.43.
25. Forgie, James W., and Nemeth, Alan G. An efficient packetized voice/data network using statistical flow control. ICC-77, Chicago, ILL., (June 1977), 38:2-44 - 38:2-48.

26. Fuchs, E., and Jackson, P.E. Estimates of distributions of random variables for certain computer communications traffic. Comm. ACM 13, 12 (Dec. 1970), 752-757.
27. Fuhrmann, Justin Jr. A glance at the electronic switching world. Telecommunications 10, 2 (Feb. 1976), 41-46.
28. Garen, Eric R., and LaZar, Les. Microprocessors in telecommunications. Telecommunications 10, 4 (Apr. 1976), 43-48.
29. Garlick, Lawrence L., Rom, Raphael, and Postel, J. Reliable host-to-host protocols: Problems and techniques. Proc. of Fifth Data Communications Symposium, Snowbird, UT., (Sep. 27-29, 1977), 4-58 - 4-65.
30. Gerla, M., and Chou, W. Flow control strategies in packet switched computer networks. NTC-74, San Diego, CA., (Dec. 2-4, 1974), 1032-1037.
31. Gitman, I., Frank, H., Occhiogrosso, B., and Hsieh, W. Issues in integrated network design. ICC-77, Chicago, ILL., (June 1977), 38:1-36 - 38:1-43.
32. Gross, Donald, and Harris, Carl M. Fundamentals of Queuing Theory, John Wiley and Sons, New York, NY., 1974.
33. Herrman, Jeff. Flow control in the ARPA network. Computer Networks 1, 1 (June 1976), 65-76.
34. Hirota, K., Kato, J., and Yoshida, Y. A design of packet switching system. ICCC-74, Stockholm, Sweden, (Aug. 12-14, 1974), 151-162.
35. Hovey, Richard B. Packet-switched networks agree on standard interface. Data Communications, (May/June 1978), 25-39.
36. Hsieh, W., Gitman, I., and Occhiogrosso, B. Design of hybrid-switched networks for voice and data. ICC-78, Toronto, Canada, (June 4-7, 1978), 20.1.1-20.1.9.
37. Itoh, Kazuo, and Kato, Takao. An analysis of traffic handling capacity of packet switched and circuit switched networks. Proc. of Datacomm 73, Third Data Communications Symposium, St. Petersburg, FL., (Nov. 1973), 29-37.
38. Jackson, James R. Networks of waiting lines. Operations Research 5, 4 (Aug. 1957), 518-521.

39. James, Richard T., and Muench, Paul E. AT&T facilities and services. Proc. of IEEE 60, 11 (Nov. 1972), 1397-1407.
40. Jenny, Christian, Kummerle, Karl, and Burge, Helmut. Network node with integrated circuit/packet-switching capabilities. Rept. RZ-720, IBM Research, (Aug. 1975), 1-61.
41. Jenny, Christian, and Kummerle, Karl. Distributed processing within an integrated circuit/packet-switching node. IEEE Trans. on Comm. Com-24, 10 (Oct. 1976), 1089-1100.
42. Jilek, P., Siemens, A.G., and Munchen, F.R. Flow control in computer networks. ICCC-74, Stockholm, Sweden, (Aug. 12-14, 1974), 151-162.
43. Jueneman, R.R., and Kerr, G.S. Explicit path routing in communications networks. ICCC-76, Toronto, Canada, (Aug. 3-6, 1976), 340-342.
44. Kahn, Robert E. Resource-sharing computer communications networks. Proc. of IEEE 60, 11 (Nov. 1972), 1397-1407.
45. Kahn, Robert E., and Crowther, William R. Flow control in a resource-sharing computer network. IEEE Trans. on Comm. Com-20, 3 (June 1972), 539-545.
46. Keyes, Neil, and Gerla, Mario. Hybrid packet and circuit switching. Telecommunications 12, 7 (July, 1978), 65-71.
47. Keyes, Neil, and Gerla, Mario. Report on experience in developing a hybrid packet and circuit switched network. ICC-78, Toronto, Canada, (June 4-7, 1978), 20.6.1-20.6.6.
48. Klein, D.M., and Frankel, S. A hybrid circuit and packet-switching system. NTC-76, Dallas, TX., (Dec. 1976), 7:3-1 - 7:3-6.
49. Kleinrock, Leonard. Queuing Systems, Volume 2: Computer Applications, John Wiley and Sons, New York, NY., (1976), 314-340.
50. Kleinrock, Leonard, and Naylor, William E. On measured behavior of the ARPA network. NTC-74, San Diego, CA., (Dec. 2-4, 1974), 767-780.
51. Korachi, T., Matsui, Y., and Asano, J. Network planning for a common use computer communication network. ICCC-76, Toronto, Canada, (Aug. 3-6, 1976), 379-382.

52. Kummerle, K. Multiplexor performance for integrated line and packet-switched traffic. ICCC-74, Stockholm, Sweden, (Aug. 12-14, 1974), 239-247.
53. Lam, Simon S. Store-and-forward buffer requirements in a packet switching network. IEEE Trans. on Comm. Com-24, 4 (April 1976), 394-403.
54. McAuliffe, Daniel J. An integrated approach to communications switching. ICC-78, Toronto, Canada, (June 4-7, 1978), 20.4.1-20.4.5.
5. McDonald, John C. Local digital switching - a successful new technology. Telecommunications 10, 4 (April, 1976), 43-48.
56. McQuillan, John M., and Walden, David C. Some considerations for a high performance message-based interprocess communication system. Proc. of ACM SIGCOMM/SIGOPS Interprocess Communications Workshop, Santa Monica, CA., (March 24-25, 1975), 77-86.
57. Miyahara, Hideo, and Hasegawa, Toshiharu. Integrated switching with variable frame and packet. ICC-78, Toronto, Canada, (June 4-7, 1978), 20.3.1-20.3.5.
58. Miyahara, Hideo, Hasegawa, Toshiharu, and Teshigawara, Yoshimi. A comparative evaluation of switching methods in computer communications networks. ICC-75, San Francisco, CA., (June 16-18, 1975), 6-3 - 6-10.
59. Naylor, T.H., Balintfy, J.L., Burdick, D.C., and Chu, K. Computer Simulation Techniques, John Wiley and Sons, New York, NY., 1966.
60. Occhiogrosso, B., Gitman, I., Hsieh, W., and Frank, H. Performance analysis of integrated switching communications systems. NTC-77, Los Angeles, CA., (Dec. 5-7, 1977), 12:4-1 - 12:4-13.
61. Opderbeck, Holger. Problems in the design of control procedures for computer networks. Computer Communication Review 5, 2 (April 1975), 1-7.
62. Opderbeck, Holger, and Kleinrock, Leonard. The influence of control procedures on the performance of packet-switched networks. NTC-74, San Diego, CA., (Dec. 2-4, 1974), 810-817.
63. Ornstein, Severo M., and Walden, David C. The evolution of a high performance modular packet-switch. ICC-75, San Francisco, CA., (June 16-18, 1975), 6-17 - 6-21.

64. Pack, Charles D. The effects of multiplexing on a computer-communications system. Comm. ACM 16, 3 (March 1973), 161-168.
65. Pack, Charles D. The output of an M/D/1 queue. Operations Research 23, 4 (July-August 1975), 750-760.
66. Paoletti, L.M. AUTODIN. Computer Communications Networks, Noordhoff International Publication, Leyden, The Netherlands, (1975), 345-372.
67. Pennotti, Michael C., and Schwartz, Mischa. Congestion control in store and forward tandem links. IEEE Trans. on Comm. Com-23, 12 (Dec. 1975), 1434-1443.
68. Pitroda, S.G. A review of telecommunications switching concepts - part two. Telecommunications 10, 3 (March 1976), 24-30.
69. Port, E., Kummerle, K., Rudin, H., Jenny, C., and Zafiropulo, P. A network architecture for the integration of circuit and packet switching. ICCC-76, Toronto, Canada, (Aug. 3-6, 1976), 505-514.
70. Pouzin, Louis. Cigale, the packet switching machine of the cyclades computer network. Proc. of IFIP Congress, Toronto, Canada, (Aug 8-12, 1977), 155-159.
71. Pouzin, Louis. Flow controls in data networks - methods and tools. ICCC-76, Toronto, Canada, (Aug. 3-6, 1976), 467-474.
72. Pouzin, Louis. Presentation and major design aspects of the cyclades computer network. Proc. of Datacomm 73, Third Data Communications Symposium, St. Petersburg, FL., (Nov. 1973), 80-87.
73. Price, W.L. Simulation of packet-switching networks controlled on isarithmic principles. Proc. of Datacomm 73, Third Data Communications Symposium, St. Petersburg, FL., (Nov. 1973), 44-49.
74. Pyke, T.N., and Blanc, R.P. Computer networking technology - a state of the art review. Computer 6, 8 (Aug. 1973), 12-19.
75. Ricci, Fred J. The state of the art and control of packet switches and networks. ICC-75, San Francisco, CA., (June 16-18, 1975), 6-1 - 6-3.

76. Rich, Marc A., and Schwartz, Mischa. Buffer sharing in computer-communication network nodes. IEEE Trans. on Comm. Com-25, 9 (Sep. 1977), 958-970.
77. Rinde, J. TYMNET I: An alternative to packet technology. ICCC-76, Toronto, Canada, (Aug 3-6, 1976), 268-273.
78. Roberts, Lawrence G. Development of packet switching networks worldwide. Telecommunications 10, 10 (Oct. 1976), 28-32.
79. Rosner, Roy D. A digital data network concept for the Defense Communications Agency. NTC-73, Atlanta, GA., (Nov. 26-28, 1973), 22C-1 - 22C-6.
80. Rosner, Roy D. Large scale network design considerations. ICCC-74, Stockholm, Sweden, (Aug. 12-14, 1974), 189-197.
81. Rosner, Roy D. Packet switching and circuit switching: A comparison. NTC-75, New Orleans, LA., (Dec. 1-3, 1975), 42-1 - 42-7.
82. Rosner, Roy D., and Springer, Ben. Circuit and packet switching - a cost and performance tradeoff study. Computer Networks 1, 1 (June 1976), 7-26.
83. Ross, Myron C. System engineering of integrated voice and data switches. ICC-78, Toronto, Canada, (June 4-7, 1978), 20.5.1-20.5.4.
84. Ross, Myron C., Tabbot, Arthur C., and Waite, John A. Design approaches and performance criteria for integrated voice/data switching. Proc. of IEEE 65, 9 (Sep. 1977), 1283-1295.
85. Rudin, Harry. On alternate-routing in circuit-switched data networks. Proc. of IFIP Congress, Toronto, Canada, (Aug. 8-12, 1977), 321-326.
86. Rudin, Harry. On routing and "delta routing": A taxonomy and performance comparison of techniques for packet-switched networks. IEEE Trans. on Comm. Com-24, 1 (Jan. 1976), 43-59.
87. Rudin, Harry. Studies on the integration of circuit and packet switching. ICCC-78, Toronto, Canada, (June 4-7, 1978), 20.2.1-20.2.7.

88. Sanders, Ray W. Comparing networking technologies. Datamation, (July 1978), 88-93.
89. Schaupp, Donald E. Survey of Computer Networks, Research Report, Texas A&M University, 1976.
90. Schmitz, H.G., Saxton, T.L., Huang, C.C., and White, J.A. Application of associative processing techniques to an integrated voice/data switched network. Rept. AD-A040636, NTIS, June 1976.
91. Shannon, Robert E. Systems Simulation - The Art and Science, Prentice-Hall, Inc., Englewood Cliffs, NJ., 1975.
92. Sullivan, Neil. TYMNET - maintenance considerations in a very large network. Proc. of Fifth Data Communications Symposium, Snowbird, UT., (Sep. 27-29, 1977), 3-1 - 3-3.
93. Thurber, Kenneth J. Circuit Switching Technology: A state-of-the-art survey. Compcon-78, Washington, DC., (Sep. 5-8, 1978), 116-124.
94. Tomita, Shuji, Yukimatsu, Ken-Ichi, Mizuno, Toshiro, and Miyaho, Noriharu. Some aspects of time-division data switch design. Proc. of IEEE 65, 9 (Sep. 1977), 1295-1304.
95. Wecker, Stuart. DECNET: A building block approach to network design. NIC-76, Dallas, TX., (Nov. 1976), 7.5-1 - 7.5-4.
96. Wong, Benedict, Giffin, Walter, and Disney, Ralph L. Two finite M/M/1 queues in tandem: A matrix solution for the steady state. Opsearch 14, 1 (1977), 1-18.
97. Yeh, Leang P. Digital telecommunications networks. Telecommunications 11, 8 (Aug. 1977), 21-25.
98. Yeh, Leang P. Telecommunication transmission media - part one. Telecommunications 10, 4 (April 1976), 51-56.
99. Zafiropulo, P. Flexible multiplexing for networks supporting line-switched and packet-switched data traffic. ICCC-74, Stockholm, Sweden (Aug. 12-14, 1974), 517-523.
100. Zafiropulo, P., Port, E., and Kummerle, K. Extension of a circuit-switched user/network interface to packet-switching. ICCC-76, Toronto, Canada, (Aug. 3-6, 1976), 515-522.

APPENDIX A

Appendix A contains a description of simulator tables, users guide, and a listing of the network simulator program used in this research. Each table is defined and its layout described. The users guide details the parameters and table building requirements. Since the program is written in FORTRAN, and the modules designed to be self-documenting, a user should have minimal difficulty using or modifying the simulator.

1. Users guide

The size of the COMMON area determines the storage requirements for the network to be simulated. There are no simulator error controls governing network configuration or size. The user has the flexibility of configuring a network topology and adjusting COMMON storage accordingly.

Each input parameter is contained in a PARAM table entry.

a. Description of Input Parameters

- (1) Contains the total number of packet and circuit switch nodes.
- (2) Establishes the total number of HDX links in the network.
- (3) Contains the number of time slots per HDX link.
- (4) Determines the number of slots in each frame which a data packet requires.
- (5) A constant which establishes the number of frames per second and service rate for data.
- (6) A constant which permits the user to establish the switching delay through each circuit switch.
- (7) Contains the voice transaction arrival rate.
- (8) Contains the data transaction arrival rate.
- (9) Holds the user specified start run time.
- (10) Holds the user specified end run time.
- (11) This parameter is a saturation level indicator. When the generation of packets exceeds this threshold, channel and queue table entries

are zeroed out. Use of this parameter is in conjunction with steady-state determination.

- (12) Establishes how many bits comprise a time slot. It is used in conjunction with PARAM entries two through four to determine the data departure rate.
- (13) Information only - it is the capacity of the T link.
- (14) Contains the service rate for voice transactions.
- (15) Establishes the average number of packets per data transaction. It is used by the geometric generation process.
- (16) Information only - it indicates how many bits per packet.
- (17) A dynamic time indicator used by the simulator when queue storage is exceeded. From this entry, the user knows how far the simulation progressed before storage was exceeded.

b. Table building specification

The user must initialize DESTAB, DSTALT, NODCHL, and SEEDTB. The formats and order of input can be determined from the program listing and Fig. 13.

2. Common Variables in the Program

All tables are contained in COMMON storage. Tables

A2-1 through A2-16 are necessary to integrate the functional modules. Each table contains a description of its use and composition.

3. Listing of the Simulator

Table A2-1. Parameter (PARAM) [X]

PARAM is a singly indexed system parameter table where X refers to the particular user input parameter. The table entries are:

1	Total number of nodes
2	Total number of channels
3	Number of slots per HDS link
4	Ratio of packet to voice slots
5	Frame time duration
6	Fixed time routing delay per node
7	Circuit switch voice arrival rate
8	Packet switch transaction arrival rate
9	Starting time for run (milliseconds)
10	Ending time for run (milliseconds)
11	Nodal packet saturation level
12	Voice Digitization Rate
13	Link capacity
14	Circuit switch service time
15	Packet size - number of bits per packet
16	Number of packets per message
17	System error run time (milliseconds)

Table A2-2. Event (EVTBL) [Node, Entry]

This table maintains a record of the next event occurrence at each node. The [Node, Entry] table entries are:

	1	2	3	4	5
1					
2					
3					
.					
.					
.					
PARAM (1)					

<u>Entry</u>	<u>Definition</u>
(1)	Time (milliseconds)
(2)	Type 1, 2, 3, 4
	1 = Class II arrival
	2 = Class II departure
	3 = Class I arrival
	4 = Class I departure
(3)	Message length
(4)	Final destination
(5)	Queue address

Table A2-3. Destination (DESTAB) [Node, Dest]

This table contains the primary channels between all node pairs [Node, Dest] and zeroes elsewhere. If the source node is a packet node, the [Node, Dest] entry contains the directly connected circuit switch node vice primary channel.

[illegible]

Table A2-4. Channel Table (CHANTB) [Channel, Entry]

There is one channel table per HDX link between circuit switch nodes. Each channel table contains PARAM (3) rows.

	1	2	3	4	5	6	7	8	9	10	11
1											
2											
3											
.											
.											
.											
PARAM (3)											

<u>Entry</u>	<u>Definition</u>
(1)	Final destination
(2)	Time channel active
(3)	Time channel available
(4)	Number of voice channels for this transaction
(5)	Destination number of intermediate node
(6)	Cumulative time channel in use
(7)	Source node for this transaction
(8)	Queue address
(9)	Cumulative transactions
(10)	Cumulative packets
(11)	Cumulative voice calls

Table A2-5 Queue (QUEUE) [Node, Entry]

Each queue table entry pertains to a packet group arrival at this node. This table only exists for packet switch nodes. The number of entries is specified by the user in the COMMON area.

	1	2	3	4	5
1										
2										
3										
.										
.										
.										
PARAM (1)										

<u>Entry</u>	<u>Definition</u>
(1)	Priority
(2)	Arrival time
(3)	Departure time
(4)	Message length
(5)	Final destination

Table A2-6. Call Queue (CALLQ) [Knode, Entry]

This table maintains an entry for departing calls at each circuit switch node.

	1	2	3	1	2	3	.	.	.
1									
2									
3									
.									
.									
.									
PARAM (1)									

<u>Entry</u>	<u>Definition</u>
(1)	Departure time
(2)	Final destination
(3)	Channel pointer

Table A2-7. Cumulative Time (CUMTIM) [Node, Entry]

This table maintains a running count of packet delays in seconds at all packet switches.

	1	2	3	4	5	6	7	8	9	10	11	12	13
1													
2													
3													
.													
.													
.													
PARAM (1)/2													

Entry	Definition
(1)	< .1 sec
(2)	< .2 sec
(3)	< .3 sec
(4)	< .4 sec
(5)	< .5 sec
(6)	< .6 sec
(7)	< .7 sec
(8)	< .8 sec
(9)	< .9 sec
(10)	< 1 sec
(11)	< 2 sec
(12)	< 5 sec
(13)	< 5 sec

Table A2-8. Calls Accepted/Rejected (CALLS) [Knode, Entry]

This table contains the number of voice calls accepted, rejected, and still in the network at each circuit switch node at run time end.

	1	2	3
1			
2			
3			
PROGRAM (1)/2			

Entry	Definition
(1)	Calls accepted
(2)	Calls rejected
(3)	Calls in system

Table A2-9. Link (LINKTB) [Node, Dest]

This is a dynamic working table containing a channel address pointer for each active node destination connection. This pointer is necessary to reduce table look-ups. Diagonal elements are unused and set to zero.

	1	2	3	.	.	.	PARAM (1)
1							
2							
3							
.							
.							
.							
PARAM (1)							

Table A2-10. Seed (SEEDTB) [Node, Distribution]

This table contains the seeds used to generate the arrival and departure times, message length, voice service rate, and destination number. It is built during initialization from inputted user specified seeds, but changed by the simulator as each seed is used.

	1	2	3	4
1				
2				
3				
.				
.				
.				
PARAM (1)				

<u>Entry</u>	<u>Definition</u>
(1)	Seed 1 - arrival
(2)	Seed 2 - departure
(3)	Seed 3 - geometric message length
(4)	Seed 4 - destination

Table A2-11. Link Availability (NLINES) [Channel]

This table is a working table that maintains a running count of available slots for each channel.

1	PARAM (3)
2	
3	
.	
.	
.	
PARAM (2)	

Table A2-12. Queue Entry Count (QENT) [Node]

This table maintains a running total of available queue entries for each packet node, and departures for each circuit node. It is used to reduce the number of queue table look-ups.

1	Entry Count
2	
3	
.	
.	
.	
PARAM (1)	

Table A2-13. Channel Connectivity (NODCHL) [Link]

This table is built during initialization by inputted system parameters. It contains the intermediate node to which each HDX link is connected.

1	Intermediate node
2	
3	
.	
.	
.	
PARAM (2)	

Table A2-14. Alternate Channel (ALTCH) [Channel]

This table is a working table which reflects which channel (primary/alternate) to use to route this transaction.

1	ALT	BIT	1 = USE	ALT
2			0 = USE	PRIMARY
3				
.				
.				
.				
PARAM (2)				

Table A2-15. Circuit Switch Arrivals (CSARV) [Knode, Entry]

This table contains information relating to each Class I arrival at a circuit switch node.

	1	2	3
1			
2			
3			
.			
.			
.			
PARAM (1)/2			

<u>Entry</u>	<u>Definition</u>
(1)	Time of arrival
(2)	Time of departure
(3)	Final destination

Table A2-16. Alternate Destination (DSTALT) [Node, Dest]

This table contains the alternate channels between all node pairs [Node, Dest]. Diagonal elements are set to zero.

	1	2	3	PARAM (1)
1				
2				
3				
.				
.				
.				
PARAM (1)				

```

*****
C THIS IS THE DRIVER-IT BUILDS USER OPINED TABLES,
C INITIALIZES ACTIVITY AT EACH NODE, AND EMPLOYS A RIGHT
C DO LOOP, CALLING THE EVENT MODULE UNTIL THE RUN TIME
C SPECIFICATION IS EXCEEDED. AT THIS POINT THE STATISTICS
C SUBROUTINE IS CALLED, FOLLOWED BY PROGRAM TERMINATION.
*****
C
C IMPLICIT INTEGER (A-S)
C ALL COMMON ENTRIES ARE USER VARIABLES.
C
COMMON /AREA1/ EVTAB(10,5),DESTAB(10,10),DSTALT(10,10),
1PARAM(17),CHANTB(576,11),DUEIF(5,1639),CALLD(5,400),
2CUMTIM(5,13),CALLS(5,3),LINKTB(10,10),SEEDTR(10,4),NLINEP(12),
3CUMT(12),NODCHL(12),ALICH(12),ESARV(5,3),
4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
5ROUT(12)
1073 FORMAT(4(I2,1X),2(I10,1X),2(I5,1X),17,1X,13,1X,14,1X,12)
1004 FORMAT(10I2)
1005 FORMAT(4(I5,1X))
1006 FORMAT(12I2)
C READ PARAM TABLE ENTRIES
READ(5,1003) (PARAM(I),I=1,16)
ALLDST=PARAM(1)
NODST=ALLDST/2
PARAM(17)=0
C READ SEEDTR ENTRIES
READ(5,1005) ((SEEDTR(I,J),J=1,4),I=1,ALLDST)
NCHNLS=PARAM(2)
C READ PRIMARY ROUTING CONFIGURATION.
READ(5,1004) ((DESTAB(I,J),J=1,ALLDST),I=1,ALLDST)

```

```

C READ ALTERNATE ROUTING CONFIGURATION.
  READ(5,1004)((DSTALT(I,J),J=1,ALLOST),I=1,ALLOST)
C READ CHANNEL -NODE RELATIONSHIPS.
  READ(5,1006)(NOCCHL(I),I=1,NCHNLS)
C INIT AVAIL SLOTS PER CHANNEL TABLE
  DO 40 I=1,NCHNLS
    NLINES(I)=PARAM(3)
  40 CONTINUE
  CLASS=2
C CREATE AN EVENT TABLE ENTRY FOR EACH NODE.
  DO 10 I=1,ALLOST
    IF(I.GT.NDEFT)CLASS=1
    CALL NEWMSG(I,CLASS)
    CALL NUEVNT(I,CLASS)
  10 CONTINUE
C START THE SIMULATION.
  20 CALL EVENT
C IF TIME LIMIT UP GO PRINT STATISTICS.
  IF(PARAM(9).LT.PARAM(10)) GO TO 20
  PARAM(10)=PARAM(9)
  PARAM(9)=PARAM(17)
  CALL STATS
C STOP THE SIMULATION.
  STOP
  END
  BLOCK DATA
C *****
C THE DATA BLOCK MAKES IT CONVENIENT FOR THE USER
C TO INITIALIZE HIS STORAGE AREAS.
C *****

```

```

      IMPLICIT INTEGER (A-S)
      COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALT(10,10),
     1PARAM(17),CHANTB(576,11),OUEUE(5,1000),CALLO(5,200),
     2CUMTIN(5,13),CALLS(5,3),LINKTB(10,10),SEEDTR(10,4),NLLINES(12),
     3ICNT(10),NODCHL(12),ALCH(12),CSARV(5,3),
     4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
     5ROUT(12)
      DATA EVTBL,CHANTB, OUEUE/0*0,6336*0,9000*0/
      DATA CALL2,CUMTIN,CALLS,LINKTR/100*0,15*0,100*0/
      DATA ICNT,ALCH,CSARV,NODLOD,DSTLOD/77*0/
      DATA DSTCNT,CUMLOD,CUMCNT/75*0/
      END
      SUBROUTINE NEWMSG(NODE,CLASS)

```

```

*****
C THIS ROUTINE GENERATES VOICE AND PACKET ARRIVALS.
C INFORMATION RELATING TO EACH ARRIVAL RESULTS IN
C A QUEUE ENTRY BEING BUILT. IF THE CURRENT LOAD EXCEEDS
C A USER SPECIFIED STEADY-STATE LOAD, CHANNEL TABLE
C STATISTICAL GATHERING ENTPIES ARE ZEROED OUT.
*****

```

```

      IMPLICIT INTEGER (A-S)
      REAL*4 ALOG
      REAL*4 RN
      COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALT(10,10),
     1PARAM(17),CHANTB(576,11),OUEUE(5,1000),CALLO(5,200),
     2CUMTIN(5,13),CALLS(5,3),LINKTB(10,10),SEEDTR(10,4),NLLINES(12),
     3ICNT(10),NODCHL(12),ALCH(12),CSARV(5,3),
     4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5)

```

```

DATA FLAG/1/
C KNODE IS USED WITH CIRCUIT SWITCH NODES TO ALLOW
C PROPER SUBSCRIPTING TO OCCUR OF THE CIRCUIT SWITCH TABLES.
  IF (CLASS.EQ.1) KNODE=NODE-(PARAM(1)/2)
  IY=0
  RN=0.0
  STIME=PARAM(9)
  IX=SEEDTB(NODE,3)
  CALL RANDOM(IX,IY,RN)
  SEEDTB(NODE,3)=IY
  DEST=PARAM(1)*RN+1
C KEEP GENERATING A NEW NODE UNTIL ONE DIFFERENT FROM
C THE SOURCE IS FOUND.
  IF (DEST.EQ.NODE) GO TO 10
  IF (CLASS.EQ.2) GO TO 20
C THIS IS A CS DEST
  IF (DEST.LE.(PARAM(1)/2)) GO TO 10
  CSARV(KNODE,3)=DEST
  IX=SEEDTB(NODE,1)
  CALL RANDOM(IX,IY,RN)
  SEEDTB(NODE,1)=IY
C GENERATE ARRIVAL TIME FOR CIRCUIT SWITCH TRANSACTION.
  ARV=(60000*(-1.0/PARAM(7)*ALOG(RN)))+PARAM(9)
  IX=SEEDTB(NODE,2)
  CALL RANDOM(IX,IY,RN)
  SEEDTB(NODE,2)=IY
  DEP=-1000.0*PARAM(14)*ALOG(RN)+ARV
  CSARV(KNODE,1)=ARV
  CSARV(KNODE,2)=DEP
  QCNT(NODE)=QCNT(NODE)+1
100 RETURN
170
20  IF (DEST.GT.(PARAM(1)/2)) GO TO 10
    X=0.0

```

```

XLANDA=PARAM(3)
CALL POISSN(XLANDA,X,NODE)
NMSG=X
IX=SEEDTR(NODE,1)
CALL RANDOM(IX,IY,RN)
SEEDB(NODE,1)=IY
APV=0
LIMIT=,CNT(NODE)
ITOP=PARAM(13)-4
DO 40 I=2,ITOP,5
  IF(QUEUE(NODE,(I-1)).EQ.0) GO TO 40
  IF(QUEUE(NODE,(I-1)).EQ.100) GO TO 40
  IF(QUEUE(NODE,(I-1)).EQ.99999) GO TO 40
  IF(QUEUE(NODE,I).LE.KEY) GO TO 40
  KEY=QUEUE(NODE,I)
  LIMIT=LIMIT-1
  IF(LIMIT.EQ.0) GO TO 50
  CONTINUE
40  IF(KEY.GT.STIME) STIME=KEY
  3  GENERATE ARRIVAL TIME FOR DATA TRAFFIC.
  APV=(1000*(-1.0*ALOG(RN)))+STIME
  2  4  GENERATE NUM OF PACKETS
  XPROB=PARAM(16)/100.0
  LENGTH=0
  NM=0
  IF(NMSG.EQ.0) NMSG=1
  DO 25 K=1,NMSG
    CALL GEOM(XPROB,NUM,NODE)
    LENGTH=LENGTH+NUM
  CONTINUE
  IF(LENGTH.EQ.0) LENGTH=1
  OSP=LENGTH*PARAM(5)+APV
  ITOP=PARAM(13)

```

```

10 30 I=1,ITOP,6
   IP(JUSTE(NODE,I),ND,0)=I
   C BUILD UP THE DATA QUEUE ENTRY.
   JUSTE(NODE,I)=1
   JUSTE(NODE,(I+1))=ABV
   JUSTE(NODE,(I+2))=DEP
   JUSTE(NODE,(I+3))=LENGTH
   JUSTE(NODE,(I+4))=DEST
   JUSTE(NODE,(I+5))=NM303
   C CARRY NODE COUNTERS.
   CUMCNT(NODE,DEST)=CUMCNT(NODE,DEST)+NM303
   NODLOD(NODE,DEST)=NODLOD(NODE,DEST)+LENGTH
   NODLOD(NODE,1)=NODLOD(NODE,1)+LENGTH
   NODLOD(NODE,2)=NODLOD(NODE,2)+LENGTH
   NODLOD(NODE,3)=NODLOD(NODE,3)+NM303
   CUMLOD(NODE,DEST)=CUMLOD(NODE,DEST)+LENGTH
   CUMCNT(NODE,DEST)=CUMCNT(NODE,DEST)+NM303
   IF(I,GT,(PARAM(13)-12)) GO TO 140
   IF(FLAG.EQ.0) GO TO 100
   C WE HAVE REACHED STEADY-STATE. MUST INITIALIZE COUNTERS.
   IP(NODLOD(NODE,I),GP,PARAM(11)) GO TO 110
   70 TO 100
   CONTINUE
   80 TO 100
   110 FLAG=0
      NDEST=PARAM(1)/2
      ALDEST=PARAM(1)
      ITOP=PARAM(2)*PARAM(3)
      90 130 I=1,ITOP
         CHANTB(I,6)=0
         CHANTB(I,9)=0
         CHANTB(I,10)=0
         CHANTB(I,11)=0

```

```

130 CONTINUE
DO 140 I=1,NDEST
DO 140 J=1,13
CUMTIN(I,J)=0
CONTINUE
DO 150 I=1,NDEST
DO 150 J=1,2
CALLS(I,J)=0
CONTINUE
PAPAM(17)=PAPAM(9)
GO TO 100
WRITE(6,2001)
ITOP=PARAM(13)-6
DO 130 I=1,ITOP,6
IEND=I+5
WRITE(6,2002) (QUEUE(NODE,J),J=I,IEND)
CONTINUE
ITOP=PARAM(2)
DO 80 K=1,ITOP
JCHNL=PARAM(3)*(K-1)+1
IEND=JCHNL+PARAM(3)-1
DO 80 L=JCHNL,IEND
WRITE(6,1005) K
WRITE(6,1003) (CHANTR(L,M),M=1,11)
CONTINUE
WRITE(6,2003) (EVTBL(J,K),J=1,5),K=1,5)
PARAM(10)=PARAM(9)
PARAM(9)=0
CALL STATS
STOP
1003 FORMAT(' ',11(I9,1X))
1005 FORMAT('0','CHANNEL ',I2)
2001 FORMAT('0','QUEUE SIZE EXCEEDED-NEWMSG 30')

```

```

2002 FORMAT(' ',5(I6,1X))
2003 FORMAT(' ',5(I6,1X))
      END
      SUBROUTINE POISSN(YLAMDA,X,NODE)
      *****
      C THIS IS THE POISSON ARRIVAL GENERATOR.
      *****
      C
      IMPLICIT INTEGER (A-S)
      REAL*4 RN
      REAL*4 EXP
      COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALT(10,10),
      1PARAM(17),CHANTB(576,11),QUEUF(5,1800),CALLO(5,200),
      2CUMTIM(5,13),CALLS(5,3),LINKTB(10,10),SEEDTB(10,4),NLINE(12),
      3DCNT(10),NODCHL(12),ALTCH(12),CSARV(5,3),
      4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
      5ROUT(12)
      X=0.0
      T=EXP(-XLAMDA)
      T1=1.0
      IY=0
      IX=SEEDTB(NODE,2)
      CALL RANDOM(IX,IY,RN)
      SEEDTB(NODE,2)=IY
      T1=T1*RN
      IF (T1-T) 4,7,7
      K=X+1.0
      GO TO 4
      RETURN
      END

```

```

SUBROUTINE GEOM(XPROB,NUM,NODE)
C*****
C THIS C THIS ROUTINE GENERATES THE NUMBER OF PACKETS
FOR A DATA TRANSACTION.
C*****
C*****
IMPLICIT INTEGER (A-S)
REAL*4 ALOG17
COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALI(10,10),
1PARAM(17),CHANTB(576,11),CHUFE(5,1600),CALLQ(5,200),
2CUMT14(5,13),CALLS(5,3),LINKTB(10,10),SEEDTB(10,4),NLINES(12),
3CUMT(10),NODCHL(12),ALICH(12),CSAFV(5,3),
4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
5ROUT(12)
REAL*4 RN
NUM=0
IV=0
1J IX=SEEDTB(NODE,4)
CALL RANDOM(IX,IY,RN)
SEEDTB(NODE,4)=IY
IF(RN.LE.XPROB)GO TO 20
NUM=NUM+1
GO TO 10
2J RETURN
END
SUBROUTINE RANDOM(IX,IY,RN)
C*****
C THIS IS THE PANDOM NUMRRR GENERATOR.

```

```

C*****
C
      IMPLICIT INTEGER (A-C)
      IY=IX*65539
      IF(IY)3,4,4
      IY=IY+2147483647+1
      RN=IY
      RN=RN*.4656611E-9
      IX=IY
      RETURN
      END
      SUBROUTINE NUEVNT(NODE,CLASS)
C*****
C
C THIS ROUTINE IS RESPONSIBLE FOR SELECTING THE NEXT
C ACTIVITY AT A NODE. ONCE AN EVENT IS SELECTED
C (ARRIVAL OR DEPARTURE), INFORMATION PERTAINING TO IT
C IS PLACED IN THE EVENT TABLE ENTRY FOR THAT NODE.
C*****
C
      IMPLICIT INTEGER (A-S)
      COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALT(10,10),
      1PARAM(17),CHANTB(576,11),CUEUE(5,1000),CALLQ(5,200),
      2CUMTIM(5,13),CALLS(5,3),LINKTB(10,10),SEEDTR(10,4),NLINES(12),
      3QCNT(10),NODCHL(12),ALTC(12),CSARV(5,3),
      4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
      5ROUT(12)
      IDELAY=0
      IYESNO=0
      LIMIT=QCNT(NODE)

```

```

C OPTIMIZE IF THIS IS A PACKET OR CIRCUIT NODE.
  IF(CLASS.EQ.2)GO TO 110
C JUST GET NEW CS ENTRY
  KNODE=NODE-(PARAM(1)/2)
C SET KEY TO MAX VALUE.
  DEP=999999
C NOW SEARCH FOR AN EVENT SMALLER THAN THE KEY.
C SEARCH CALLQ TABLE LOOKING FOR SOONEST ACTIVITY.
  DO 20 I=1,150,4
    IF(CALLQ(KNODE,I).EQ.0)GO TO 20
    IF(CALLQ(KNODE,I).GE.DEP)GO TO 15
    DEP=CALLQ(KNODE,I)
    INDEX=I
  15  LIMIT=LIMIT-1
    IF(LIMIT.EQ.0)GO TO 5
  20  CONTINUE
  5   IF(DEP.EQ.999999)GO TO 25
    IF(CSARV(KNODE,1).GE.DEP)GO TO 30
C NOW PLACE CIRCUIT SWITCH INFORMATION IN EVENT TABLE.
  25  EVTBL(NODE,1)=CSARV(KNODE,1)
    EVTBL(NODE,2)=3
    EVTBL(NODE,3)=CSARV(KNODE,2)
    EVTBL(NODE,4)=CSARV(KNODE,3)
    GO TO 100
  30  EVTBL(NODE,1)=DEP
    EVTBL(NODE,2)=4
    EVTBL(NODE,4)=CALLQ(KNODE,(INDEX+1))
    EVTBL(NODE,5)=INDEX
    GO TO 100
  110 KEY=999999
    ITOP=PARAM(13)-3
    DO 40 I=3,ITOP,6
      IF QUEUE(I) EQUAL ZERO THEN SKIP IT.

```

```

C IF QUEUE(1) EQUAL 100 SKIP IT BECAUSE IT IS THE
C THE RETURN PATH FOR A CONNECTION.
  IF(QUEUE(NODE,(I-2)).EQ.0)GO TO 40
  IF(QUEUE(NODE,(I-2)).EQ.100)GO TO 40
  IF(QUEUE(NODE,I).GE.KFY)GO TO 41
  IF(QUEUE(NODE,(I-2)).NE.999999)GO TO 41
  KEY=QUEUE(NODE,I)
  FLAG=2
  INDEX=I-2
  LIMIT=LIMIT-1
  IF(LIMIT.EQ.0)GO TO 45
  CONTINUE
C THIS LOGIC CHECKS TRAFFIC THAT HAS BEEN IN QUEUE FOR
C SOME TIME TO SEE IF IT CAN BE SENT YET.
  45  PRIORITY=0
      KK=4
      DO 55 KKK=1,5
      K=KK-KKK
      LIMIT=QCNT(NODE)
      IF(PRIORITY.NE.0)GO TO 75
      ITOP=PARAM(13)-4
      DO 50 I=2,ITOP,6
      IF(QUEUE(NODE,(I-1)).EQ.0)GO TO 50
      IF(QUEUE(NODE,(I-1)).EQ.999999)GO TO 46
      IF(QUEUE(NODE,(I-1)).EQ.100)GO TO 46
      IF(QUEUE(NODE,I).GE.KFY)GO TO 46
      IF(QUEUE(NODE,(I-1)).NE.K)GO TO 46
      DEST=QUEUE(NODE,(I+3))
C HAVE FOUND THE OLDEST TRAFFIC, CHECK FIRST PATH.
      PASS=1
      CALL ROUTE(NODE,DEST,IYESNO,IDELAY,CLASS,PASS)
      IF(IYESNO.EQ.1)GO TO 46
      IF(IYESNO.PQ.2)GO TO 46

```

```

C 404 CHECK RETURN PATH.  IF OK THEN IT CAN BE SENT,
PASS=2
CALL ROUTE(DEST,NODE,IYESNO,IDELAY,CLASS,PASS)
IF (IYESNO.EQ.2) GO TO 46
IF (IYESNO.EQ.1) GO TO 46
30  KEY=QUEUE(NODE,I)
    FLAG=1
    INDEX=I-1
    PRIOR=K
    LIMIT=LIMIT-1
    IF (LIMIT.LE.0) GO TO 55
40  CONTINUE
50  CONTINUE
55  IF (KEY.NE.999999) GO TO 75
    IF (KKK.LT.5) GO TO 75
    LIMIT=QCNT(NODE)
    FLAG=1
    ITOP=PARAM(13)-5
    KEY=QUEUE(NODE,2)
    DO 35 I=1,ITOP,6
C 35  SOONEST DEPARTURE.
    IF (QUEUE(NODE,I).EQ.0) GO TO 35
    IF (QUEUE(NODE,I).LT.6) GO TO 36
35  CONTINUE
30  KEY=QUEUE(NODE,(I+1))
    INDEX=I
    DO 60 I=1,ITOP,6
C 60  SOONEST ARRIVAL.
    IF (QUEUE(NODE,I).EQ.0) GO TO 60
    IF (QUEUE(NODE,I).EQ.100) GO TO 65
    IF (QUEUE(NODE,I).EQ.9999) GO TO 65
    IF (QUEUE(NODE,(I+1)).GT.KEY) GO TO 65
30  IF (QUEUE(NODE,I).LT.5) GO TO 65

```

```

      QUEUE(NODE,I)=1
      LIMIT=LIMIT-1
      IF (LIMIT.EQ.0) GO TO 75
      GO TO 60
35    KEY=QUEUE(NODE,(I+1))
      INDEX=I
      IF (QUEUE(NODE,(I+1)).LE.PARAM(3)) GO TO 66
      GO TO 75
90    CONTINUE
      C SELECT FROM ARRIVAL OR DEPARTURE, AND PLACE INFORMATION
      C RELATING TO IT ON EVENT TABLE.
75    EVTL(NODE,1)=QUEUE(NODE,(INDEX+1))
      IF (FLAG.EQ.2) EVTL(NODE,1)=QUEUE(NODE,(INDEX+2))
      EVTL(NODE,2)=FLAG
      EVTL(NODE,3)=QUEUE(NODE,(INDEX+3))
      EVTL(NODE,4)=QUEUE(NODE,(INDEX+4))
      EVTL(NODE,5)=INDEX
100   RETURN
      END
      SUBROUTINE STATS
C*****
C THIS ROUTINE IS SELF DOCUMENTING.
C IT IS RESPONSIBLE FOR OUTPUT GENERATION OF STATISTICAL
C INFORMATION.
C*****
      IMPLICIT INTEGER (A-S)
      COMMON /AREA1/ EVTL(10,5),DESTAB(10,10),DSTALT(10,10),
      1PARAM(17),CHANTB(576,11),QUEUE(5,1800),CALLC(5,200),
      2CUMT1(5,13),CALLS(5,3),LINKTB(10,10),SFEDTR(10,4),NLINES(12),

```

```

3CONT(10),NOCHL(12),ALICH(12),CSAGN(5,3),
4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
5ROUT(12)
C WRITE TOP OF PAGE INFORMATION.
WRITE(6,2020)((FVTL(J,K),J=1,5),K=1,5)
WRITE(6,2020)
WRITE(6,2011)
WRITE(6,2012)
WRITE(6,2011)(PARAM(I),I=1,16)
ILIM=PARAM(2)
DO 10 I=1,ILIM
WRITE(6,2002)I
WRITE(6,2003)
K=1
C CALCULATE THE UTILIZATION FOR THIS CHANNEL.
JCHNL=(PARAM(3)*(I-1))+1
ITOP=JCHNL+PARAM(3)-1
UTIL=0.
PSENT1=0
DO 20 J=JCHNL,ITOP
IF(CHANTB(J,4).EQ.0)GO TO 80
CHANTB(J,6)=CHANTB(J,6)+PARAM(10)-CHANTB(J,2)
PSENT=CHANTB(J,10)
PSENT1=PSENT1+PSENT
UTIL=1.0*CHANTB(J,6)/(PARAM(10)-PAFAM(4))
UTIL1=UTIL+UTIL
IVoice=CHANTB(J,11)
C WRITE OUT CHANNEL INFORMATION.
WRITE(6,2004)K,PSENT,UTIL,IVoice
K=K+1
20 CONTINUE
UTIL1=UTIL1/PARAM(3)
WRITE(6,2017)PSENT1,UTIL1

```

```

10  CONTINUE
C  WRITE OUT PACKET NODE INFORMATION.
  WRITE(6,2005)
  WRITE(6,2006)
  ITOP=PARAM(1)/2
  DO 30 I=1,ITOP
    WRITE(6,2007) I, (CUMTIM(I,K), K=1,13)
  CONTINUE
30  CONTINUE
  WRITE(6,2013)
  IUP=PARAM(13)-4
  DO 60 I=1,ITOP
    DO 90 J=2,IUP,6
      IF (QUEUE(I,J).LE.PARAM(10)) GO TO 90
      NODLOD(I,1)=NODLOD(I,1)-1
      JPTR=QUEUE(I, (J+3))
      DSTLOD(I,JPTR)=DSTLOD(I,JPTR)-10
      DSTCNT(I,JPTR)=DSTCNT(I,JPTR)-1
    CONTINUE
  CONTINUE
  WRITE(6,2014) I, (NODLOD(I,K), K=1,3)
60  CONTINUE
  WRITE(6,2015)
  ALDST=PARAM(1)/2
  DO 70 I=1,ALDST
    DO 70 J=1,ALDST
      IF (I.EQ.J) GO TO 70
      WRITE(6,2016) I,J,DSTLOD(I,J),DSTCNT(I,J),CUMLOD(I,J),CUMCNT(I,J)
    CONTINUE
70  CONTINUE
C  WRITE OUT CIRCUIT NODE STATISTICS.
  WRITE(6,2009)
  WRITE(6,2009)
  DO 40 I=1,ITOP
    K=PARAM(1)/2+I
    ITOT=CALLS(I,1)+CALLS(I,2)

```

```

ILOST=CALLS(I,2)
UTIL=).0
IF (ITOT.EQ.0) GO TO 50
UTIL=1.0*CALLS(I,2)/ITOT
IKEEP=CALLS(I,3)
WRITE(6,2010)K,ITOT,ILOST,UTIL,IKEEP
20 CONTINUE
RETURN
2000 FORMAT('1',40X,'SYSTEM PARAMETERS')
2001 FORMAT('1',1X,3(12,4X),11,5X,12,MS',12,1' MS',3X,12,
1' MIN',3X,12,SEC',2X,15,1' MS',16,1' MS',17,1X,
215,'KBS',17,3X,13,SEC',14,8,2X,12)
2002 FORMAT('1',40X,'CHANNEL',2X,12,2X,'UTILIZATION')
2003 FORMAT('1',SLOT',5X,'PACKETS SENT',6X,'PER CENT UTILIZATION',
15X,'NUMBER OF VOICE CALLS')
2004 FORMAT('1',2X,12,9X,18,22X,F4.2,2CX,16)
2005 FORMAT('1',40X,'PACKET NODE STATISTICS')
2006 FORMAT('0',NODE',5X,'DELAY (SECS)',4X,'<.1',4X,'<.2',4X,
1'<.3',4X,'<.4',4X,'<.5',4X,'<.6',4X,'<.7',4X,'<.8',4X,
2'<.9',4X,'< 1',4X,'< 2',4X,'< 5',4X,'> 5')
2007 FORMAT('1',2X,12,16X,17,12(16,1X))
2008 FORMAT('1',40X,'CIRCUIT SWITCH NODE STATISTICS')
2009 FORMAT('0',NODE',5X,'TOTAL CALLS',5X,'CALLS LOST',9X,
1'BLOCKING',5X,'CALLS IN SYSTEM')
2010 FORMAT('1',2X,12,11X,15,13X,F4.2,10X,15)
2011 FORMAT('0',1X,'NODES LINKS SLOTS RATIO SLOT NODE CS',6X,
1'PS MSG START TIME END TIME PACKET VDR PAGES',
2'Q SIZE CS PACKET PACKETS')
2012 FORMAT('1',25X,'TIME DELAY ARRIVAL ARRIVAL',21X,
1'LOADING',19X,'SERVICE SIZE PER MSG')
2013 FORMAT('1',NODE CURRENT PACKET LOAD CUMULATIVE PACKET',
1'LOAD CUMULATIVE TRANSACTIONS')
2014 FORMAT('1',2X,12,10X,110,13X,110,14X,110)

```

```

2015  FORMAT(' ', 'NODE DEST CURRENT PACKET LOAD CURRENT ',
1' TRANSACTION LOAD CUMULATIVE PACKET LOAD CUMULATIVE',
2' TRANSACTION LOAD')
2016  FORMAT(' ', '2X,12,3X,12,10X,110,15X,110,13X,110,18X,110)
2017  FORMAT(' ', 'TOTAL PACKETS SENT ',17,' TOTAL LINK UTILIZATION',
12X,F4.2)
2020  FORMAT(' ',5(I6,1X))
      END
      SUBROUTINE EVENT
C*****
C THIS ROUTINE SCANS THE EVENT TABLE ENTRIES LOOKING FOR
C THE NEXT SYSTEM EVENT. IT THEN BRANCHES TO THE
C APPROPRIATE ROUTINE TO SERVICE THE EVENT.
C*****
      IMPLICIT INTEGER (A-S)
      COMMON /AREA1/ EVTBL(10,5), DESTAB(10,10), DSTALT(10,10),
1PAPAY(17), CHANTB(576,11), QUEUE(5,1800), CALLO(5,200),
2CUMTIV(5,13), CALLS(5,3), LINKTB(10,10), SPEDTB(10,4), NLINES(12),
3SCNT(10), NODCHL(12), ALTCH(12), CSABV(5,3),
4NODLOD(5,3), DSTLOD(5,5), DSTCNT(5,5), CUMLOD(5,5), CUMCNT(5,5),
5ROUT(12)
      NOROUT=0
      REST=999999
C FIND SOONEST ACTIVITY
      ALLOST=PAPAY(1)
      DO 10 I=1,ALLOST
      IF (EVTBL(I,1).EQ.0) GO TO 10
      IF (EVTBL(I,1).GE.REST) GO TO 10
      REST=EVTBL(I,1)

```

```

      NODE=I
      CONTINUE
      C NOW PROCESS EVENT OCCURRENCE TYPE
      C 1=PS ARR,2=PS DEP,3=CS ARR,4=CS DEP
      KEY=EVTL(NODE,2)
      GO TO (100,200,300,400),KEY
      C PROCESS PS ARRIVAL
      100 CLASS=2
      CALL ARRIVE(NODE,CLASS)
      150 CALL VENTSS(NODE,CLASS)
      155 CALL VEVENT(NODE,CLASS)
      160 RETURN
      C PROCESS PS DEPARTURE
      200 CLASS=2
      CALL DEPART(NODE,CLASS)
      GO TO 155
      C NOW PROCESS CS ARRIVAL
      300 CLASS=1
      CALL ARRIVE(NODE,CLASS)
      GO TO 150
      C CS DEPARTURE
      400 CLASS=1
      CALL DEPART(NODE,CLASS)
      GO TO 155
      END
      SUBROUTINE ROUTE(LNODE,KDEPT,IYESNO,IDELAY,CLASS,PASS)
      C*****
      C THIS ROUTINE IS USED TO FIND A ROUTE THROUGH THE NETWORK.
      C IT IS CALLED TWICE FOR EACH CIRCUIT SWITCH ROUTE. NTRACE IS
      C A TABLE USED TO INSURE NO LOOPS OCCUR IN THE ROUTINE
      C PROCESS. IYESNO SIGNIFIES WHETHER OR NOT A GOOD

```

ROUTE WAS FOUND.

```

*****
IMPLICIT INTEGER (A-S)
DIMENSION NTRACE(10)
COMMON /APEA1/ EVTBL(10,5),DESTAB(10,10),DSTALL(10,10),
1PARAM(17),CHANS(576,11),SHEVE(5,1400),CALLS(5,200),
2COUNT(5,13),CALLS(5,3),LINKTS(10,10),SPEEDT(10,4),NLINE(12),
3CNT(10),NODCHL(12),ALCH(12),CSARV(5,3),
4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
5CNT(12)
1ROUT=PARAM(7)
IF(PASS.EQ.2)GO TO 80
DO 55 I=1,ROUT
  ALCH(I)=0
  ROUT(I)=0
CONTINUE
DO 99 I=1,ROUT
  NTRACE(I)=0
CONTINUE
NODE=LNODE
DEST=KDEST
INODE=NODE
NCOUNT=1
ITYPE=0
IDELAY=0
RATIO=1
ALT=0
IF(CLASS.EQ.2)RATIO=PARAM(4)
NTRACE(NCOUNT)=INODE
IYESNO=0

```

```

C IS THIS A RS ROUTE
  IF (CLASS.EQ.2) INODE=DESTAB(INODE,DEST)
  IF (CLASS.EQ.1) GO TO 10
  IDELAY=PARAM(6)
  NCOUNT=NCOUNT+1
  NTRACE(NCOUNT)=INODE
  C DOES PATH ALREADY EXIST FOR THIS TRANSACTION
  ICHAN=LINKTB(INODE,DEST)
  IF (ICHAN.EQ.0) GO TO 10
  ITYPE=1
  10  IF (INODE.EQ.DEST) GO TO 60
  ICHAN=DESTAB(INODE,DEST)
  25  IF (RATIO.GT.NLINES(ICHAN)) GO TO 20
  C THERE ARE SLOTS AVAILABLE
  ROUT(ICHAN)=RATIO+ROUT(ICHAN)
  IF ((NLINES(ICHAN)-ROUT(ICHAN)).LT.0) GO TO 50
  ALT=0
  INODE=NOOCHL(ICHAN)
  DO 30 K=1,NCOUNT
  IF (INODE.EQ.NTRACE(K)) GO TO 50
  30  CONTINUE
  NCOUNT=NCOUNT+1
  NTRACE(NCOUNT)=INODE
  C ADD DELAY TIME FOR SIGNALLING THROUGH THIS NODE.
  IDELAY=PARAM(6)+IDELAY
  IF (DESTAB(INODE,DEST).EQ.0) GO TO 60
  30 TO 10
  20  IF (ALT.EQ.1) GO TO 50
  ALT=1
  IF (ALTCH(ICHAN).EQ.1) GO TO 50
  ALTCH(ICHAN)=1
  ICHAN=OSTALT(INODE,DEST)
  GO TO 25

```



```

IF (QADDR.EQ.0) GO TO 50
INDEX(1)=QADDR
PASS=2
CALL GET(DEST,NODE,CLASS,QADDR,CHPIS,PASS)
IF (QADDR.EQ.0) GO TO 50
INDEX(2)=QADDR

5 REMOVE FIRST HALF OF CONNECTION.
PASS=1
CALL REMOVE(NODE,DEST,CLASS,PASS)
6 REMOVE RETURN PATH FOR TRANSACTION.
PASS=2
CALL REMOVE(DEST,NODE,CLASS,PASS)
7 DETERMINE IF TRANSACTION WAS A CIRCUIT OF DATA TRANSACTION.
IF (CLASS.EQ.1) GO TO 20
KNODE=NODE
KDEST=DEST
DO 10 I=1,2
DEST=KDEST
NODE=KNODE
DCNT(NODE)=DCNT(NODE)-1
K=INDEX(I)
8 UPDATE PACKET NODE STATISTICS.
IF (DSTLOC(NODE,DEST)-QUEUE(NODE,(K+3)).LT.0) GO TO 30
DSTLOC(NODE,DEST)=DSTLOC(NODE,DEST)-QUEUE(NODE,(K+3))
DSTCNT(NODE,DEST)=DSTCNT(NODE,DEST)-QUEUE(NODE,(K+5))
NDLOC(NODE,1)=NDLOC(NODE,1)-QUEUE(NODE,(K+3))
ITOP=K+5
9 PURGE THE Q ENTRIES FOR DATA.
DO 40 M=K,ITOP
QUEUE(NODE,M)=0
CONTINUE
40
KNODE=DEST

```

```

10 KDEST=NODE
15 CONTINUE
20 PARAM(9)=EVTRL(LNODE,3)
25 RETURN
30 KNODE=NODE-(PARAM(1)/2)
35 KDEST=DEST-(PARAM(1)/2)
40 REMOVE CIRCUIT SWITCH TRANSACTION FROM CALLQ TABLE.
45 CALLQ(KNODE,(INDEX(1)))=0
50 CALLQ(KDEST,(INDEX(2)))=0
55 KNODE=CALLQ(KDEST,(INDEX(2)+3))-(PARAM(1)/2)
60 CALLS(KNODE,3)=CALLS(KNODE,3)-1
65 CALL N'EVENT(DEST,CLASS)
70 GO TO 35
75 K=EVITBL(LNODE,5)
80 ITOP=K+5
85 DO 60 M=K,ITOP
90 QUEUE(LNODE,M)=0
95 CONTINUE
100 GO TO 40
105 END
110 SUBROUTINE REMOVE(LNODE,KDEST,CLASS,PASS)
115 *****
120 *****
125 *****
130 *****
135 *****
140 *****
145 *****
150 *****
155 *****
160 *****
165 *****
170 *****
175 *****
180 *****
185 *****
190 *****
195 *****
200 *****
205 *****
210 *****
215 *****
220 *****
225 *****
230 *****
235 *****
240 *****
245 *****
250 *****
255 *****
260 *****
265 *****
270 *****
275 *****
280 *****
285 *****
290 *****
295 *****
300 *****
305 *****
310 *****
315 *****
320 *****
325 *****
330 *****
335 *****
340 *****
345 *****
350 *****
355 *****
360 *****
365 *****
370 *****
375 *****
380 *****
385 *****
390 *****
395 *****
400 *****
405 *****
410 *****
415 *****
420 *****
425 *****
430 *****
435 *****
440 *****
445 *****
450 *****
455 *****
460 *****
465 *****
470 *****
475 *****
480 *****
485 *****
490 *****
495 *****
500 *****
505 *****
510 *****
515 *****
520 *****
525 *****
530 *****
535 *****
540 *****
545 *****
550 *****
555 *****
560 *****
565 *****
570 *****
575 *****
580 *****
585 *****
590 *****
595 *****
600 *****
605 *****
610 *****
615 *****
620 *****
625 *****
630 *****
635 *****
640 *****
645 *****
650 *****
655 *****
660 *****
665 *****
670 *****
675 *****
680 *****
685 *****
690 *****
695 *****
700 *****
705 *****
710 *****
715 *****
720 *****
725 *****
730 *****
735 *****
740 *****
745 *****
750 *****
755 *****
760 *****
765 *****
770 *****
775 *****
780 *****
785 *****
790 *****
795 *****
800 *****
805 *****
810 *****
815 *****
820 *****
825 *****
830 *****
835 *****
840 *****
845 *****
850 *****
855 *****
860 *****
865 *****
870 *****
875 *****
880 *****
885 *****
890 *****
895 *****
900 *****
905 *****
910 *****
915 *****
920 *****
925 *****
930 *****
935 *****
940 *****
945 *****
950 *****
955 *****
960 *****
965 *****
970 *****
975 *****
980 *****
985 *****
990 *****
995 *****

```

```

      IMPLICIT INTEGER (A-S)
      COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALT(10,10),
1PARAM(17),CHANTB(576,11),QUEUF(5,1800),CALLQ(5,200),
2CUMTIM(5,13),CALLS(5,3),LINKTB(10,10),SECTB(10,4),NLINES(12),

```

```

3QCNT(10),NODCHL(12),ALTC(12),CSARV(5,3),
4NODLOD(5,3),DSTLOP(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
5RPUT(12)
  NODE=LNODE
  DEST=KDEST
  IP(CLASS.EQ.1)KNODE=NODE-(PARAM(1)/2)
  FLAG=1
  ILIM=1
  IF(CLASS.EQ.2)ILIM=PARAM(4)
  INODE=NODE
  IF(CLASS.EQ.2)INODE=DESIAB(NODE,DEST)
  ICHAN=DESIAB(INODE,DEST)
  IF(INODE.EQ.DEST)GO TO 40
  FLAG=1
35  IF(FLAG.EQ.0)ICHAN=DSTALT(INODE,DEST)
  JCHNL=PARAM(3)*(ICHAN-1)+1
  ITOP=JCHNL+PARAM(3)-1
  I=JCHNL-1
  DO 15 J=1,ILIM
    JCHNL=I+1
    DO 20 I=JCHNL,ITOP
      IF(CHANTB(I,7).NE.NODE)GO TO 20
    C CHECK FOR MATCH BY SOURCE, DEST, AND TIME.
      IF(CHANTB(I,1).NE.DEST)GO TO 20
      IF(PASS.EQ.2)GO TO 90
      IF(CHANTB(I,3).NE.EVTBL(NODE,1))GO TO 20
    C HAVE FOUND CHANNEL MATCH, PURGE CHANNEL ENTRIES.
    CHANTB(I,6)=CHANTB(I,6)+(CHANTB(I,3)-CHANTR(I,2))
    CHANTB(I,2)=0
    CHANTB(I,3)=0
    CHANTB(I,1)=0
    NLINES(ICHAN)=NLINES(ICHAN)+1

```

```

      CHANTB(I,4)=0
      INODE=CHANTR(I,5)
      CHANTB(I,5)=0
      30 TO 70
      IF (CHANTB(I,3).EQ.PVTR1(DEST,1)) GO TO 60
      CONTINUE
      IF (FLAG.EQ.0) GO TO 80
      FLAG=0
      50 TO 15
      70 IF (CLASS.EQ.1) CHANTB(I,11)=CHANTB(I,11)+1
      IF (CLASS.EQ.1) GO TO 15
      PTR=CHANTB(I,3)
      CHANTB(I,9)=CHANTR(I,9)+QUEUE(NODE,(PTR+5))
      CHANTB(I,10)=CHANTB(I,11)+QUEUE(NODE,(PTR+3))
      15 CONTINUE
      C KEEP LOOPING THROUGH THIS CHANNEL TABLE UNTIL ALL ENTRIES
      C ARE FOUND AND PURGED.
      30 ICHAN=DESTAB(INODE,DEST)
      IF (DESTAB(INODE,DEST).EQ.0) GO TO 40
      50 TO 10
      50 WRITE(6,1000) LNODE, KDEST
      40 IF (CLASS.NE.2) GO TO 50
      INODE=DESTAB(NODE,DEST)
      50 RETURN
1000 FORMAT('0','ERROR IN REMOVE',2(1X,I2))
      END
      SUBROUTINE ARRIVE(LNODE,CLASS)
      C*****
      C
      C THIS ROUTINE IS RESPONSIBLE FOR ARRIVAL OF A DATA/VOICE
      C TRANSACTION AT THIS NODE. IT IS THE DRIVER-WITH RESPONSIBILITY
      C FOR GETTING A ROUTE, UPDATING CHANNEL TABLES, AND

```

C GENERATING PACKET DELAY INFORMATION.

```

*****
IMPLICIT INTEGER (A-S)
DIMENSION LDELAY(12)
COMMON /AREA1/ EVTBL(10,5),DESTAB(10,10),DSTALT(10,10),
1PARAM(17),CHANTR(576,11),CHEFE(5,1800),CALLQ(5,200),
2CUTIN(5,13),CALLS(5,3),LINKTB(10,10),SEPDTB(10,4),NLINES(12),
3CUNT(10),NODCHL(12),ALICH(12),CGARV(5,3),
4NODLOD(5,3),DSTLOD(5,5),DSTCNT(5,5),CUMLOD(5,5),CUMCNT(5,5),
5ROUT(12)
DATA LDELAY/101,201,301,401,501,601,701,801,901,1001,2001,5001/
LDELAY=0
FLAG=1
NODE=LNODE
IF (EVTBL(NODE,1).GT.PARAM(9)) PARAM(9)=EVTBL(NODE,1)
STIME=PARAM(9)
DEST=EVTBL(NODE,4)
INDXQ=EVTBL(NODE,5)
IYESNO=0
PASS=1
CALL ROUTE(NODE,DEST,IYESNO,LDELAY,CLASS,PASS)
C BRANCH IF CS TRANSACTION
IF (CLASS.EQ.1)GO TO 30
C VALUE OF IYESNO 0-NO EXISTING ROUTE, PATH AVAILABLE
1-EXISTING ROUTE AVAILABLE, NO PATH
2-NO EXISTING ROUTE AVAILABLE, NO PATH
3-EXISTING PATH AVAILABLE, PATH AVAILABLE
IF (IYESNO.EQ.2)GO TO 10
IF (IYESNO.EQ.3)GO TO 20
IF (IYESNO.EQ.0)GO TO 20
IF (IYESNO.EQ.1)GO TO 10

```

```

C NOW ADD DELAY TO CUM TIME TABLE
75 IF (FLAG.EQ.1) GO TO 80
   NODL=DEST
   DO 70 I=1,12
   IF (IDELAY.GT.LDELAY(I)) GO TO 70
C ADD DELAY TIME FOR THESE PACKETS.
   CUMTIM(NODE,I)=CUMTIM(NODE,I)+NPCKTS
   IF (FLAG.EQ.0) GO TO 140
   FLAG=0
   NPCKTS=NPCKTS/5+1
   IDELAY=50
   GO TO 75
70 CONTINUE
   CUMTIM(NODE,13)=CUMTIM(NODE,13)+NPCKTS
   IF (FLAG.EQ.0) GO TO 180
   FLAG=0
   GO TO 75
180 LTOP=PARAM(2)
   DO 50 L=1,LTOP
   ALCH(L)=0
50 CONTINUE
   RETURN
C NO PATH AVAILABLE
10 INDEX=EVTL(NODE,5)
   QUEUE(NODE,INDEX)=QUEUE(NODE,INDEX)+1
   GO TO 180
C MUST BUILD NEW PS PATH
20 IDELAY=0
C NOW CHK 2ND HALF OF PDX LINE
   PASS=2
   CALL ROUTE(DEST,NODE,IYESNO,IDELAY,CLASS,PASS)
   IF (IYESNO.EQ.2) GO TO 10
   IF (IYESNO.EQ.1) GO TO 10

```

```

IF(EVTBL(NODE,1).NE.STIME)GO TO 25
IDELAY=IDELAY+STIME-EVTBL(NODE,1)
25 INDEX=EVTBL(NODE,5)
NPCKTS=QUEUE(NODE,(INDEX+3))
NMSGSGS=QUEUE(NODE,(INDEX+5))
C UPDATE QUEUE ENTRIES WITH INFORMATION PLACED IN CHANNEL TABLES.
QUEUE(NODE,INDEX)=399999
QUEUE(NODE,(INDEX+1))=QUEUE(NODE,(INDEX+1))+IDELAY
QUEUE(NODE,(INDEX+2))=QUEUE(NODE,(INDEX+2))+IDELAY
ITOP=PARAM(13)
DO 90 I=1,ITOP,6
C FIND A FREE QUEUE ENTRY FOR RETURN PATH.
IF(QUEUE(DEST,I).NE.0)GO TO 30
QUEUE(DEST,I)=100
QUEUE(DEST,(I+1))=QUEUE(NODE,(INDEX+1))
QUEUE(DEST,(I+2))=QUEUE(NODE,(INDEX+2))
QUEUE(DEST,(I+3))=NPCKTS/5+1
QUEUE(DEST,(I+4))=NODE
C ASSUME RETURN PATH CARRIES SOME TRAFFIC. ADD THIS FACTOR TO TABLES.
QUEUE(DEST,(I+5))=NMSGSGS/5+1
CNT(DEST)=CNT(DEST)+1
DSTLOD(DEST,NODE)=DSTLOD(DEST,NODE)+NPCKTS/5+1
DSTCNT(DEST,NODE)=DSTCNT(DEST,NODE)+NMSGSGS/5+1
NODLOD(DEST,1)=NODLOD(DEST,1)+NPCKTS/5+1
NODLOD(DEST,2)=NODLOD(DEST,2)+NPCKTS/5+1
NODLOD(DEST,3)=NODLOD(DEST,3)+NMSGSGS/5+1
CUMLOD(DEST,NODE)=CUMLOD(DEST,NODE)+NPCKTS/5+1
CUMCNT(DEST,NODE)=CUMCNT(DEST,NODE)+NMSGSGS/5+1
GO TO 35
90 CONTINUE
C GO ACTUALLY UPDATE CHANNEL TABLES.
35 PASS=1
CALL UPDATE(NODE,DEST,CLASS,IDELAY,PASS)

```

```

PASS=2
CALL UPDATE(DEST,NODE,CLASS,IO-LAY,PTAB)
IF (CLASS.EQ.2) GO TO 75
GO TO 140
C C3 ARRIVAL
30  NDEST=PARAM(1)/2
    KNODE=NODE-NDEST
    IF (YESNO.EQ.1) GO TO 40
    IF (YESNO.EQ.2) GO TO 41
    YESNO=0
    IDELAY=0
    PASS=2
    CALL ROUTE(DEST,NODE,YESNO,IDELAY,CLASS,PASS)
    IF (YESNO.EQ.1) GO TO 40
    IF (YESNO.EQ.2) GO TO 40
    CALLS(KNODE,1)=CALLS(KNODE,1)+1
    CALLS(KNODE,3)=CALLS(KNODE,3)+1
    GO TO 35
40  CALLS(KNODE,2)=CALLS(KNODE,2)+1
    GO TO 140
END
SUBROUTINE UPDATE(LNODE,LDEST,CLASS,IDELAY,PASS)
C*****
C THIS ROUTINE UPDATES CHANTB ENTRIES FOR THE ROUTE SELECTED.
C*****
    IMPLICIT INTEGER (A-S)
    COMMON /AREA1, EVTAB(10,5),DESTAB(10,10),DSTALI(10,10),
     1PARAM(17),CHANTR(576,11),QUEIF(5,1800),CALLQ(5,200),
     2CONTIM(5,14),CALLS(5,1),LINKTB(10,10),SEEDTR(10,4),NLINES(12),

```

```

3 CONT(10), NOCHL(12), ALCH(12), CSARY(5,3),
+NODED(5,3), DSTLOD(5,5), DSTCNT(5,5), CUNLOD(5,5), CUPCNT(5,5),
5 PRINT(12)
NODE=LNODE
DESN=LDEST
INDEX=EVTHL(NODE,5)
INODE=NODE
RATIO=1
IF (CLASS.EQ.2) RATIO=PARAM(4)
FLAG=1
IF (CLASS.EQ.1) KNODE=INODE- (PARAM(1)/2)
IF (CLASS.EQ.2) INODE=DESTAR(NODE,DEST)
+DETERMINE IF DATA OR VOICE CONNECTION.
IF (CLASS.EQ.1) GO TO 7C
IF (PASS.EQ.2) GO TO 150
IF (INODE.EQ.DEST) GO TO 50
ICHAN=DESTAR(INODE,DEST)
IF (ALCH(ICHAN).EQ.1) GO TO 30
END ALT ROUTE
50 NLINES(ICHAN)=NLINES(ICHAN)-RATIO
JCHNL=PARAM(3)*(ICHAN-1)+1
ITOP=JCHNL+PARAM(3)-1
I=JCHNL-1
DO 15 J=1,RATIO
JCHNL=I+1
DO 20 I=JCHNL,ITOP
+ MUST FIND FREE CHANNEL IN THE LINK.
IF (CHANB(I,4).EQ.0) GO TO 25
CONTINUE
WRITE(6,204)
IF (FLAG.EQ.0) GO TO 40
IF (CLASS.EQ.1) GO TO 26

```



```

100 CONTINUE
    WRITE(6,201)
    GO TO 50
110 ATCH(ICHAN)=0
    ICHAN=OSTAT(INODE,DEST)
    GO TO 35
200 RETURN
300 DO 100 I=1,150,4
    IF CALL2(KNODE,I).EQ.0 GO TO 130
100 CONTINUE
    IF (PASS.EQ.2) GO TO 140
    LEAD= BOTH CHANNELS FOR CIRCUIT SWITCH CONNECTION.
110 CALL2(KNODE,I)=CSARY(KNODE,2)+IDELAY
    CALL2(KNODE,(I+3))=LNODE
120 CALL2(KNODE,(I+1))=LDEST
    CALL2(KNODE,(I+2))=I
    INDEX=I
    GO TO 10
130 KDEST=DEST-(PARAM(1)/2)
    CALL2(KNODE,I)=CSARY(KDEST,3)+IDELAY
    CALL2(KNODE,(I+3))=LDEST
    GO TO 170
140 FORMAT('0','ERROR IN UPDATE 16')
150 FORMAT('0','ERROR IN UPDATE 20')
    END
    SUBROUTINE GET2(LNODE,LDEST,CLASS,ADDR,CHRTB,PASS)
*****
100 THIS ROUTINE IS USED WHEN IT IS NECESSARY TO TERMINATE
200 A ROUTE. IT FINDS THE STARTING CHANNEL ADDRESS
300 FOR THE REMOVAL PROCESS.

```

```

*****
      IMPLICIT INTEGER (A-Z)
      COMMON /AEEA1/ EVTRI(10,5),CT,TT(10,1),INSTALL(10,10),
     1PARAM(17),CHANTR(5,5,10),NECODE(1,10),SALDO(0,200),
     2CUMTIV(5,10),CALLS(5,3),LINEPTR(1,10),DEFCT(10,9),NLINES(12),
     3JANT(1),SODCHL(12),ALPHA(10),DASV(1,3),
     4NODLDD(5,1),DSTLDD(5,5),DSTANT(5,5),MLX(5,5),CUMCNT(5,5),
     5JANT(12)
      FLX=1
      NODE=LNODE
      DEPT=LODEP
      INJDE=NODE
      IF (CLASS.EQ.0) INODE=DEFIAR(INODE,INST)
      ICHAN=DESTAR(INODE,DEST)
      IF (FLAG.EQ.0) ICHAN=DESTAR(INODE,DEST)
      JCHNL=PARAM(3)*(ICHAN-1)+
     1TOP=JCHNL+PARAM(3)-1
      DO 10 I=JCHNL,ITOP
     1 TRY TO MATCH UP SOURCE, DEST, AND TIME.
      IF (CHANTR(J,7).NE.NODE) GO TO 10
      IF (CHANTR(J,1).NE.DEST) GO TO 10
     20 TO 40
     30 JADDF=CHANTR(J,4)
      CHPTR=J
      GO TO 30
     40 IF (PAST.EQ.2) GO TO 40
      IF (CHANTR(J,4).EQ.EVTRI(NODE,1)) GO TO 50
     50 TO 10
     60 IF (CHANTR(J,3).EQ.EVTRI(DEST,1)) GO TO 60
     70 CONTINUE
     80 IF (FLAG.EQ.0) GO TO 70
      FLAG=0
*****

```

AD-A107 254

AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH
ANALYSIS OF FLOW BEHAVIOR WITHIN AN INTEGRATED COMPUTER-COMMUNI--ETC(U)
MAY 79 C A CLABAUGH
AFIT-CI-79-2130

F/G 17/2

UNCLASSIFIED

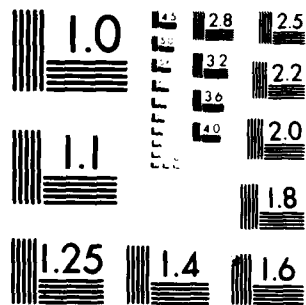
NL

3 13

30

000000

END
DATE
FILMED
12-81
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A₅

```

      GO TO 20
      C FOUND MATCH-RETURN ADDRESS.
30    RETURN
      C ERROR CONDITION-NO MATCH FOUND.
70    WRITE(6,1000)
      STIME=PARAM(9)
      WRITE(6,1001) LNODE, LDEST, PASS, STIME
      WRITE(6,1002) (EVTBL(NODE,K), K=1,5)
      ITOP=EVTBL(LNODE,5)
      ILIM=ITOP+5
      WRITE(6,1004) (QUEUE(LNODE,M), M=ITOP, ILIM)
      GO TO 30
1000  FORMAT('0','ERROR IN GETC')
1001  FORMAT('0',4(I6,1X))
1002  FORMAT('0',5(I6,2X))
1004  FORMAT('0',6(I7,1X))
      END

```

APPENDIX B

Appendix B contains a users guide and listing of the analytic model. The program is written in FORTRAN and highly portable. The program documentation correlates very closely to the analytic development.

1. Users guide

a. Description of Input Parameters

- (1) LAMDA1 - contains data transaction arrival rate per minute.
- (2) LAMDA2 - contains voice transaction arrival rate per minute.
- (3) MU1 - contains data transaction service rate per minute.
- (4) MU2 - contains voice transaction service rate per minute.
- (5) NSLOTS - establishes the channel state-space size.
- (6) MLIMIT - establishes a user specified steady-state indicator. It must be equal or greater than NSLOTS.

b. Run-Time Considerations

Because the model uses large matrices, double precision accuracy, and is compute bound, it requires considerable CPU resources. For use on the Texas A&M Amdahl computer facility, an NSLOTS of 48 required one megabyte and 50 seconds of CPU time. To keep costs to a minimum, the program must be run during level 0 operation.

2. Listing of the program

```

C *****
C
C THIS PROGRAM IMPLEMENTS A NETWORK MATHEMATICAL MODEL
C THAT DECOMPOSES AN INTEGRATED VOICE/DATA COMPUTER-COMMUNICATION
C NETWORK INTO NODAL QUEUING MODELS. EACH NODE CAN BE
C REPRESENTED USING A RATE MATRIX AND LINEAR MATRIX CONCEPTS.
C FIRST THE PROGRAM FINDS THE PROBABILITY  $\rho$  VECTOR IN THE RATE
C MATRIX. IT THEN SOLVES FOR ALL REMAINING PROBABILITY
C VECTORS IN TERMS OF  $\rho(0)$ . THE MATRIX IS THEN NORMED.
C FINALLY STEADY-STATE EFFECTIVENESS MEASURES ARE DETERMINED.
C  $\rho$  LIMIT IS A STEADY-STATE USER SPECIFIED VARIABLE USED TO
C ADJUST THE SIZE OF THE RATE MATRIX. THIS MATRIX
C ADJUSTMENT IS NECESSARY TO DETERMINE WHEN STEADY-STATE
C HAS OCCURRED BY NOTING THE DIFFERENCES IN EFFECTIVENESS
C MEASURES WITH EACH CHANGE OF  $\rho$  LIMIT.
C
C THE USER MUST INPUT VOICE AND DATA ARRIVAL AND
C DEPARTURE RATES, THE CHANNEL SIZE, AND  $\rho$  LIMIT.
C DOUBLE PRECISION ACCURACY IS REQUIRED.
C AZERO IS A(0) IN THE ALGORITHM.
C A IS A(N), N=0,1,2, ..., C-1 IN THE ALGORITHM.
C M1 IS M(1) IN THE ALGORITHM.
C MC IS M(N), N=0,1, ..., N-1
C M(PRIME) IS M(N), N=LARGE N-1
C LAMBDA IS A MATRIX OF VOICE ARRIVAL RATES
C B1 CONSISTS OF 0 H2
C
C I H1
C A(PRIME) IS A(N), N=LARGE N-1
C AC IS A(N), N=C, ..., LARGE N-2
C WHERE C IS THE CHANNEL SIZE.
C LARGE N IS A USER DATA ROUND(MLIMIT) ON THE RATE MATRIX SIZE.
C B1 = H1 IN THE ALGORITHM
C B2 = H2 IN THE ALGORITHM.

```

```

C E1, E2 ARE H1,H2 BUT M(PRIME) IS USED VICE MC TO BUILD E1,E2.
C E1,E2 BUILD B(PRIME) IN THE ALGORITHM.

```

```

C *****

```

```

      IMPLICIT REAL*4(A-H,O-Z)
      REAL*4 LAMBDA,M1,MC,MCC,MU1,MU2,LAMDA1,LAMDA2
      COMMON /AREA1/ MU1,MU2,LAMDA1,LAMDA2,N,NSLOTS,MLIMIT
      COMMON /AREA2/ A(50,50)
      DIMENSION BC(100,100),BD(100,100),S(100,100),B1(50,50)
      DIMENSION B2(50,50),E1(50,50),E2(50,50),ACC(50,50),M1(50)
      DIMENSION PROB(100,50),P(50,50)
      DIMENSION TEMP(50,50),R(100,100),VCTR(100),MC(50)
      DIMENSION AC(50,50),AMAT(50),LAMBDA(50),MCC(50)
      DIMENSION D1(50),D2(50),EIGEN(100),X(50)
      DIMENSION VCTR1(50)

```

```

C READ INPUT PARAMETERS
      READ(5,200) LAMDA1,LAMDA2,MU1,MU2,NSLOTS,MLIMIT
      WRITE(6,203) LAMDA1,LAMDA2,MU1,MU2,NSLOTS,MLIMIT
      IEND=NSLOTS-1
      N=NSLOTS+1

```

```

      NSIZE=N

```

```

      ILIM=N+1

```

```

      ITOP=2*N

```

```

      DO 725 I=1,NSLOTS

```

```

        LAMBDA(I)=LAMDA1

```

```

        M1(I)=1.0/MU1

```

```

        K=N-I

```

```

        MC(I)=1.0/(K*MU1)

```

```

        MCC(I)=1.0/(K*MU1)

```

```

      CONTINUE

```

```

      LAMBDA(N)=0.0

```

```

      MC(N)=-1.0/(NSLOTS*MU2+LAMDA1)

```

```

725

```

```

MCC(N)=-1.0/(NSLOTS*MU2)
Y1(N)=-1.0/(NSLOTS*MU2+LAMDA1)
C BUILD LAMDA AND Y1, MC MATRICES
DO 10 I=1, N
  IT=N+I
DO 10 J=1, N
  BC(I,J)=0.0
  BC(II,J)=0.0
  B1(I,J)=0.0
  B2(I,J)=0.0
  E1(I,J)=0.0
  E2(I,J)=0.0
  ACC(I,J)=0.0
  AC(I,J)=0.0
  IF(I.NE.J) GO TO 10
  BC(II,J)=1.0
10 CONTINUE
C NOW BUILD AC AND ACC
DO 20 I=1, N
  II=N+I
DO 20 J=1, N
  IF(I.NE.J) GO TO 20
  IF(I.EQ.1) GO TO 25
  AC(I,J)=-1.0*((I-1)*MU2+(N-I)*MU1+LAMDA1)
  AC(I,(J-1))=(I-1)*MU2
  ACC(I,(J-1))=(I-1)*MU2
  ACC(I,J)=-1.0*((I-1)*MU2+(N-I)*MU1)
  GO TO 20
25 AC(I,J)=-1.0*(NSLOTS*MU1+LAMDA1)
  ACC(I,J)=-1.0D0*NSLOTS*MU1
20 CONTINUE
  AC(N,N)=LAMDA1
  ACC(N,N)=0.0

```

```

C NOW BUILD B1,B2,E1,E2
DO 710 I=1,N
  II=N+I
  DO 710 J=1,N
    B1(I,J)=B1(I,J)+AC(I,J)*MC(J)
    E1(I,J)=E1(I,J)+AC(I,J)*MCC(J)
    B1(I,J)=-1.0*B1(I,J)
    E1(I,J)=-1.0*E1(I,J)
    IF(I.NE.J)GO TO 710
    B2(I,J)=B2(I,J)+LAMBDA(J)*MC(J)
    E2(I,J)=E2(I,J)+LAMBDA(J)*MCC(J)
    B2(I,J)=-1.0*B2(I,J)
    E2(I,J)=-1.0*E2(I,J)
  710 CONTINUE
C BUILD A(N),N=0
  K=0
  KK=K+1
  CALL BLDA(K,AMAT)
  CALL BLDMAT(KK,AMAT)
C START THE ITERATIVE LOOP BUILDING G1,G2,---G(C-1)
C FLIP-FLOPPING BC AND BD TO REPRESENT G1,G2,ETC.
  K=1
  KK=K+1
  CALL BLDA(K,AMAT)
  CALL BLDMAT(KK,AMAT)
  DO 715 I=1,N
    AMAT(I)=1.0/AMAT(I)
  715 CONTINUE
C BUILD G1 USING BC
C AMAT IS A SINGLE DIM ARRAY WHICH IS THE INVERSE OF M(N)
  DO 720 I=1,N
    II=N+I
    DO 720 J=ILIM,ITOP

```

```

BC(II,J)=0.0
BC(I,J)=0.0
JJ=J-N
BC(II,J)=BC(II,J)+A(I,JJ)*AMAT(JJ)
BC(II,J)=-1.0*BC(II,J)
IF(I.NE.JJ)GO TO 720
BC(I,J)=BC(II,J)+LAMBDA(JJ)*AMAT(JJ)
5C(I,J)=-1.0*BC(I,J)
720 CONTINUE
C HAVE B1 NO DO B1.....B(C-1)
MM=0
DO 730 NN=2,IEND
K=K+1
KK=K
CALL BLDMAT(KK,AMAT)
CALL BLDA(K,AMAT)
KK=K+1
CALL BLDMAT(KK,AMAT)
DO 739 I=1,N
D1(I)=LAMBDA(I)*1.0/AMAT(I)
DO 739 J=1,N
A(I,J)=A(I,J)*1.0/AMAT(J)
CONTINUE
739 DO 735 I=1,N
II=N+I
DO 735 J=1,N
JJ=N+J
IF(MM.EQ.1)GO TO 736
BD(I,J)=BC(I,JJ)
BD(II,J)=BC(II,JJ)
GO TO 735
736 BC(I,J)=BD(I,JJ)
BC(II,J)=BD(II,JJ)

```

```

735 CONTINUE
C CAN COPY PART OF S(N) TO G(N+1) DIRECTLY.
DO 740 I=1,N
  II=N+I
  DO 740 J=1,N
    JJ=N+J
    IF(MM.EQ.1) GO TO 741
    BD(I,JJ)=BC(I,J)*D1(J)
    BD(II,JJ)=BC(II,J)*D1(J)
    GO TO 740
741 5C(I,JJ)=BD(I,J)*D1(J)
    5C(II,JJ)=BD(II,J)*D1(J)
740 CONTINUE
DO 745 I=1,N
  II=N+I
  DO 745 J=1,N
    JJ=N+J
    DO 744 L=1,N
      LL=N+L
      IF(MM.EQ.1) GO TO 746
      IF((DABS(BC(I,LL)*A(L,J))) .LE. 1.0E-14) GO TO 742
      BD(I,JJ)=BD(I,JJ)+BC(I,LL)*A(L,J)
      C MUST DO MATRIX MULTIPLICATION TO GO FROM G(N) TO G(N+1)
742 IF((DABS(BC(II,LL)*A(L,J))) .LE. 1.0E-14) GO TO 744
      BD(II,JJ)=BD(II,JJ)+BC(II,LL)*A(L,J)
      GO TO 744
746 IF((DABS(BD(I,LL)*A(L,J))) .LE. 1.0E-14) GO TO 743
      BC(I,JJ)=BC(I,JJ)+BD(I,LL)*A(L,J)
743 IF((DABS(BD(II,LL)*A(L,J))) .LE. 1.0E-14) GO TO 744
      BC(II,JJ)=BC(II,JJ)+BD(II,LL)*A(L,J)
744 CONTINUE
      IF(MM.EQ.1) GO TO 747
      BD(I,JJ)=-1.0*BD(I,JJ)

```

```

30(II,JJ)=-1.0*BD(II,JJ)
GO TO 745
747 BC(I,JJ)=-1.0*BC(I,JJ)
745 BC(II,JJ)=-1.0*BC(II,JJ)
CONTINUE
*RITE(6,202)NN
IF(NN.EQ.1)GO TO 750
NN=1
GO TO 730
750 NN=0
730 CONTINUE
C HAVE FINISHED G1, - - ,G(C-1)
C MUST NOW GET EIGENVALUES FOR MATRIX OPERATOR B.
IP(NN.EQ.0)GO TO 765
DO 760 I=1,ITOP
DO 760 J=1,ITOP
BC(I,J)=BD(I,J)
CONTINUE
760
765 ISOUND=HLINIT-NSLOTS
IP(IBOUND.EQ.0)GO TO 160
C NOW GET EIGENVALUES
IPTR=1
C CALCULATE EACH QUADRATIC SOLUTION(EIGENVALUES).
DO 410 I=1,NSLOTS
K=I-1
T1=LAMDA1+K*MU2+(N-I)*MU1
T2A=(LAMDA1+K*MU2+((N-I)*MU1)**2
T2B=-4.0*LAMDA1*(N-I)*MU1
T2=T2A+T2B
T3=2.0*(N-I)*MU1
EIGEN(IPTR)=(T1+DSQRT(T2))/T3
IPTR=IPTR+1
EIGEN(IPTR)=(T1-DSQRT(T2))/T3

```

```

      IPTP=IPTP+1
410  CONTINUE
      EIGEN(ITOP-1)=1.0*LAMDA1/(NSLOTS*MU2+LAMDA1)
      EIGEN(ITOP)=0.0
      C NOW WRITE OUT EIGENVALUES
      WRITE(6,1002)
      DO 411 K=1,ITOP,8
        L=K+7
        WRITE(6,1003) (EIGEN(I),I=K,L)
411  CONTINUE
      C USING EACH EIGENVALUE L PROCPDE TO BUILD R3 AND R4
      L=1
      IFLAG=0
      C LOOPING RTN TO GET R3 - R4 EIGENVECTORS
      DO 430 N=1,N
      C APPLY EIGENVECTOR L,L=0,1,- - -, (2*NSLOTS)
      C TO MATRIX D.
      C BUILD MATRIX FOR EIGENVALUE L
      DO 440 I=1,NSLOTS
        K=I-1
        T=LAMDA1+(N-I)*MU1+K*MU2
        T1=(N-I)*MU1
        D2(I)=EIGEN(L)**2-(EIGEN(L)*T/T1)+LAMDA1/T1
        IF(I.EQ.1)GO TO 450
        D1(I)=(EIGEN(L)*MU2*K)/((N-K)*MU1)
        GO TO 440
      D1(I)=0.0
450  CONTINUE
      D1(N)=(EIGEN(L)*NSLOTS*MU2)/MU1
      D2(N)=EIGEN(L)**2-(EIGEN(L)*LAMDA1/(NSLOTS*MU2+LAMDA1))
      C D IS BUILT FOR EIGENVALUE L
      C NOW CONSTRUCT ACCOMPANYING VECTOR
      DO 460 IPTP=1,N

```

```

T=D2(IPTR)
IF(DABS(T).LE.1.0E-5) GO TO 470
VCTR(IPTR)=0.0
450 CONTINUE
C ERROR-COULD NOT ZERO EQUATION FOR EIGENVALUE.
WRITE(6,1006)
STOP
470 VCTR(IPTR)=1.0
IPTR=IPTR+1
IF(IPTR.GT.N)GO TO 465
C HAVE GOT ZERO EQUATION-BUILD EIGENVECTOR.
DO 480 I=IPTR,N
T=D1(I)*VCTR(I-1)
IF(DABS(T).LE.1.0E-13) GO TO 475
T=T/D2(I)
VCTR(I)=-1.0*T
GO TO 480
475 VCTR(I)=0.0
480 CONTINUE
DO 475 IPTR=1,N
IPTR1=N+1-IPTR
T=D2(IPTR1)
IF(DABS(T).LE.1.0E-5) GO TO 477
VCTR1(IPTR1)=0.0
C KEEP PLACING ZEROES IN EIGENVECTR.
475 CONTINUE
WRITE(6,1006)
STOP
C MUST PLACE A 1 IN THE EIGENVECTOR ACCORDING TO ALGORITHM.
477 VCTR1(IPTR1)=1.0
IPTR1=IPTR1-1
IF(IPTR1.LT.1)GO TO 485
DO 479 I=1,IPTR1

```

```

II=IPTR1+1-I
N1=D1(II+1)*VCTR1(II+1)
IP(DARS(T1),LP,1.0E-13) GO TO 478
N1=T1/D2(II)
VCTR1(II)=-1.0*T1
C BUILD NEXT EIGENVECTOR.
GO TO 473
478 VCTR1(II)=0.0
479 CONTINUE
C NOW HAVE R3 OR R4 EIGENVECTOP
485 MM=M
JJ=N
IF(IFLAG.EQ.1)MM=M+N
IP(IFLAG.EQ.1)JJ=0
C PLACE VCTR IN R3 OR R4
DO 490 I=ILIM,ITOP
KK=I-JJ
II=I-N
P(I,MM)=VCTR(II)
S(MM,II)=VCTR1(II)
490 CONTINUE
L=L+1
430 CONTINUE
IF(IFLAG.EQ.1)GO TO 510
IFLAG=1
GO TO 500
C S1,S3 ARE BUILT-BUILD S2 AND S4
C P1,R4 ARE BUILT-BUILD R1,R2,AND J MATRICES
C BE HOLDS R3J0, BC HOLDS R1P3
C R1=R3J0-B1R3
C R2=R4J1-B1R4
C TMP HOLDS B1R4, A HOLDS R4J1
C S2=J0S1, S4=J1S3

```

```

510 DO 530 I=1,N
    II=N+I
DO 530 J=1,N
    JJ=J+N
    TEMP(I,J)=0.0
    BD(I,J)=0.0
    A(I,J)=R(II,JJ)*EIGEN(JJ)
    P(I,J)=R(II,J)*EIGEN(J)
    S(I,JJ)=EIGEN(I)*S(I,J)
    S(II,JJ)=EIGEN(II)*S(II,J)
DO 530 L=1,N
    LL=L+N
    BD(I,J)=BD(I,J)+B1(I,L)*R(LL,J)
    TEMP(I,J)=TEMP(I,J)+B1(I,L)*R(LL,JJ)
530 CONTINUE
DO 540 I=1,N
DO 540 J=1,N
    JJ=N+J
    R(I,J)=P(I,J)-BD(I,J)
    R(I,JJ)=A(I,J)-TEMP(I,J)
540 CONTINUE
C RAISE J TO HLIMIT.
DO 300 I=1,ITOP
    EIGEN(I)=EIGEN(I)**(IROUND-1)
300 CONTINUE
DO 305 I=1,ITOP
DO 305 J=1,ITOP
C GET RJ
    BD(I,J)=R(I,J)*EIGEN(J)
305 CONTINUE
C FORM TRUE S.
DO 550 I=1,ITOP
DO 550 J=1,ITOP

```

```

IP(I,NE,J)GO TO 550
VCTR(I)=0.0
DO 551 L=1,ITOP
VCTR(I)=VCTR(I)+S(I,L)*R(L,J)
551 CONTINUE
550 CONTINUE
DO 555 I=1,ITOP
DO 555 J=1,ITOP
S(I,J)=S(I,J)/VCTR(I)
555 CONTINUE
DO 315 I=1,ITOP
DO 315 J=1,ITOP
R(I,J)=0.0
DO 315 L=1,ITOP
C GET RJEXP(N)S
R(I,J)=R(I,J)+BD(I,L)*S(L,J)
315 CONTINUE
DO 322 I=1,N
II=N+I
DO 322 J=1,N
S(I,J)=0.0
S(II,J)=0.0
IP(I,NE,J)GO TO 322
S(II,J)=1.0
322 CONTINUE
DO 323 I=1,N
II=N+I
C INCORPORATE B(PRIME)
DO 323 J=1,N
JJ=N+J
S(I,JJ)=E2(I,J)
S(II,JJ)=E1(I,J)
323 CONTINUE

```

```

DO 320 I=1,ITOP
II=N+I
DO 320 J=1,N
JJ=J+N
BD(I,JJ)=0.0
3D(I,J)=R(I,JJ)
DO 320 L=1,ITOP
BD(I,JJ)=SD(L,JJ)+R(I,L)*S(L,JJ)
320 CONTINUE
DO 40 I=1,ITOP
DO 40 J=1,ITOP
R(I,J)=0.0
DO 40 L=1,ITOP
R(I,J)=R(I,J)+BC(I,L)*RD(L,J)
40 CONTINUE
GO TO 180
150 DO 170 I=1,ITOP
DO 170 J=1,ITOP
R(I,J)=BC(I,J)
170 CONTINUE
180 K=0
CALL BLDA(K,AMAT)
DO 60 I=1,N
DO 60 J=1,N
C TEMP CORRELATES WITH EQUATION 5.1.10 IN ALGORITHM. A(ZERO)*
C M1(INVERSE.
TEMP(I,J)=0.0
TEMP(I,J)=TEMP(I,J)+A(I,J)*M1(J)
TEMP(I,J)=-1.0*TEMP(I,J)
60 CONTINUE
DO 70 I=1,N
DO 70 J=1,N
BC(I,J)=R(I,J)

```

```

DO 70 L=1,N
II=L+N
C MULTIPLY EQUATION 5.1.10 MATRICES.
BC(I,J)=BC(I,J)+TEMP(I,L)*R(II,J)
70 CONTINUE
DO 80 I=1,N
DO 80 J=ILIM,ITOP
BC(I,J)=R(I,J)
DO 80 L=1,N
II=L+N
BC(I,J)=BC(I,J)+TEMP(I,L)*P(II,J)
80 CONTINUE
DO 90 I=1,N
DO 90 J=1,N
P(I,J)=0.0
DO 90 L=1,N
LL=N+L
C BUILD P(ZERO) MATRIX.
P(I,J)=P(I,J)+BC(I,LL)*ACC(L,J)
90 CONTINUE
DO 91 I=1,N
DO 91 J=1,N
P(I,J)=P(I,J)+BC(I,J)*LAMBDA(J)
91 CONTINUE
DO 97 I=1,N
DO 97 J=1,N
IF(J.LE.I) GO TO 97
TMP=P(I,J)
P(I,J)=P(J,I)
P(J,I)=TMP
97 CONTINUE
P(N,1)=1.0
N=N+1

```

```

DO 110 I=1,N
X(I)=0.0
110 CONTINUE
X(N)=1.0
C MUST SOLVE P MATRIX USING GAUSSIAN ELIMINATION TO
C GET P(ZERO) VECTOR.
CALL LINSYS(P,N,X)
C BUILD 1...C-1 PROBABILITY MATRIX
K=0
CALL BLDA(K,AMAT)
C BUILD P(1) VECTOR.
DO 800 I=1,N
PROB(1,I)=X(I)
800 CONTINUE
DO 805 J=1,N
TEMP(1,J)=0.0
DO 806 L=1,N
TEMP(1,J)=TEMP(1,J)+X(L)*A(L,J)
806 CONTINUE
TEMP(1,J)=-1.0*TEMP(1,J)
805 CONTINUE
C BUILD P(2) VECTOR.
DO 810 I=1,N
PROB(2,I)=0.0
PROB(2,I)=PROB(2,I)+TEMP(1,I)*Y1(I)
810 CONTINUE
C NOW HAVE P0,P1, BUILD P2....PC-1
K=1
KK=1
C NOW LOOP THRU PROB 3-PROB C-1
DO 830 M=2,NSLOTS
IPTR=M+1
CALL BLDMAT(KK,AMAT)

```

```

      CALL BLDA(K,AMAT)
      KK=K+1
      CALL BLDMAT(KK,AMAT)
      DO 821 I=1,N
        AMAT(I)=1.0/AMAT(I)
      CONTINUE
      K=K+1
      DO 840 J=1,N
        TEMP(1,J)=PROB((M-1),J)*LAMBDA(J)
      DO 841 L=1,N
        TEMP(1,J)=TEMP(1,J)+PROB(M,L)*A(L,J)
      CONTINUE
      TEMP(1,J)=-1.0*TEMP(1,J)
      CONTINUE
      DO 845 J=1,N
        PROB(IPTR,J)=0.0
        PROB(IPTR,J)=PROB(IPTR,J)+TEMP(1,J)*AMAT(J)
      CONTINUE
      CONTINUE
      C HAVE NOW BUILT PROBABILITY MATRIX TP TO C-1.
      DO 851 J=1,N
        JJ=N+J
        PROB((MLIMIT-1),J)=0.0
        PROB(MLIMIT,J)=0.0
      DO 851 L=1,N
        C BUILD P(LARGE N-1).
        C BUILD P(LARGE N).
        PROB((MLIMIT-1),J)=PROB((MLIMIT-1),J)+X(L)*BC(L,J)
        PROB(MLIMIT,J)=PROB(MLIMIT,J)+X(L)*BC(L,JJ)
      CONTINUE
      ISTOP=MLIMIT-2
      DO 855 M=ILIM,ISTOP
        IPTR=M

```

```

DO 860 J=1,N
I=M-1
TEMP(1,J)=PROB((I-1),J)*LAMBDA(J)
C BUILD PROBABILITY MATRIX P(C) THRU P(N-2).
DO 861 L=1,N
TEMP(1,J)=TEMP(1,J)+PROB(I,L)*AC(L,J)
861 CONTINUE
TEMP(1,J)=-1.0*TEMP(1,J)
860 CONTINUE
DO 865 J=1,N
PROB(IPTR,J)=0.0
PROB(IPTR,J)=PROB(IPTR,J)+TEMP(1,J)*MC(J)
865 CONTINUE
855 CONTINUE
TOT=0.0
DO 870 I=1,MLIMIT
DO 870 J=1,N
IF (PROB(I,J).LT.0) PROB(I,J)=0.0
TOT=TOT+PROB(I,J)
870 CONTINUE
WRITE(6,1004)MLIMIT
C NORM THE PROBABILITY MATRIX.
DO 875 I=1,MLIMIT
PROB(I,50)=0.0
DO 875 J=1,N
PROB(I,J)=PROB(I,J)/TOT
PROB(I,50)=PROB(I,50)+PROB(I,J)
875 CONTINUE
DO 876 I=1,N
PROB(100,I)=0.0
DO 876 J=1,MLIMIT
PROB(100,I)=PROB(100,I)+PROB(J,I)
876 CONTINUE

```

```

DO 877 K=1,ITOP,8
L=K+7
WRITE(6,201) (PROB(100,I),I=K,L)
377 CONTINUE
DO 878 K=1,MLIMIT,3
L=K+7
WRITE(6,201) (PROB(1,50),I=K,L)
378 CONTINUE
C OBTAIN SYSTEM STEADY-STATE INFORMATION.
SYSDAT=0.0
DO 890 I=1,MLIMIT
II=I-1
I=0.0
DO 891 J=1,N
I=I+PROB(I,J)
391 CONTINUE
SYSDAT=SYSDAT+II*I
390 CONTINUE
C WRITE EXPECTED NUMBER OF DATA TRANSACTIONS IN SYSTEM.
WRITE(6,1020) SYSDAT
QDATA=0.0
DO 895 I=1,N
JJ=N-I+1
DO 895 J=JJ,MLIMIT
QDATA=QDATA+((J-JJ)*PROB(J,I))
395 CONTINUE
C WRITE EXPECTED NUMBER OF DATA TRANSACTIONS IN QUEUE.
396 WRITE(6,1021) QDATA
SYSVOC=0.0
DO 900 I=1,N
II=I-1
I=0.0
DO 901 J=1,MLIMIT

```

```

I=T+PROB(J,I)
301 CONTINUE
   SYSVOC=SYSVOC+II*T
302 CONTINUE
C WRITE EXPECTED NUMBER OF VOICE CALLS IN SYSTEM.
  WRITE(6,1022)SYSVOC
  FILVOC=0.0
  DO 905 I=1,MLIMIT
    FILVOC=FILVOC+PROB(I,N)
905 CONTINUE
C WRITE PROB. OF ALL CHANNELS USED BY VOICE.
  WRITE(6,1023)FILVOC
  T=0.0
  ALLVOC=0.0
  DO 910 I=1,MLIMIT
    II=I-1
    ALLVOC=ALLVOC+II*PROB(I,N)
    T=T+PROB(I,N)
910 CONTINUE
  IF(T.LT.1.0E-5) GO TO 911
  ALLVOC=ALLVOC/T
C WRITE DATA CONDITIONED ON ALL CHANNELS USED BY VOICE.
911 WRITE(6,1024)ALLVOC
  II=N-2
  SVOICE=0.0
  DO 915 I=1,MLIMIT
    SVOICE=SVoice+PROB(I,II)+PROB(I,NSLOTS)+PROB(I,N)
915 CONTINUE
C WRITE PROB. SYSTEM ALMOST FULL.
  WRITE(6,1025)SVoice
  EXVOC=0.0
  T=0.0
  DO 920 I=1,MLIMIT

```

```

JJ=I-1
EXVOC=EXVOC+JJ*(PROB(I,II)+PROB(I,NSLOTS)+PROB(I,N))
T=T+PROB(I,II)+PROB(I,NSLOTS)+PROB(I,N)
CONTINUE
IF(T.LT.1.0E-5) GO TO 921
EXVOC=EXVOC/T
C WRITE DATA CONDITIONED ON SYSTEM ALMOST FULL.
921 WRITE(6,1026) EXVOC
STOP
200 FORMAT(4(F8.4,1X),I2,1X,I5)
201 FORMAT(' ',8(E14.6,1X))
202 FORMAT(' ',PASS NUMBER ',I3)
203 FORMAT(' ',LAMBDA1=',F8.4,' LAMBDA2=',F8.4,' MU1=',F8.4,
1' MU2=',F8.4,' CHANNELS=',I3,' LIMIT=',I3)
205 FORMAT(' ',8(E14.5,1X))
1002 FORMAT(' ',EIGENVALUES ARE')
1003 FORMAT(' ',9(F15.8,1X))
1004 FORMAT(' ',PROBABILITY MATRIX ',I3)
1006 FORMAT(' ',ERROR AT 460')
1020 FORMAT(' ',EXPECTED DATA TRANSACTIONS IN SYSTEM',I4,E15.8)
1021 FORMAT(' ',EXPECTED DATA TRANSACTIONS IN QUEUE',I4,E15.8)
1022 FORMAT(' ',EXPECTED VOICE CALLS IN SYSTEM ',E15.8)
1023 FORMAT(' ',PROB OF ALL SLOTS USED BY VOICE ',E15.8)
1024 FORMAT(' ',DATA CONDITIONED ON FULL VOICE ',E15.8)
1025 FORMAT(' ',PROB VOICE USING MORE THAN 2-2 SLOTS ',E15.8)
1026 FORMAT(' ',DATA CONDITIONED ON VOICE >C-2 ',E15.8)
END
SUBROUTINE 3LGA(K,AMAT)
C *****
C THIS SUBROUTINE BUILDS A(N),N=0,1,2,- - ,C-1.
C

```

```

*****
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 LAMBDA, T1, MC, MCC, MU1, MU2, LAMDA1, LAMDA2
      COMMON /AREA1/ MU1, MU2, LAMDA1, LAMDA2, N, NSLOTS, MLIMIT
      COMMON /AREA2/ A(50,50)
      DIMENSION AMAT(50)
      NUM=NSLOTS-K
      DO 10 I=1, N
      DO 10 J=1, N
      A(I,J)=0.0
      CONTINUE
      DO 20 I=1, N
      DO 20 J=1, N
      IF (I.NE.J) GO TO 20
      IF (I.EQ.1) GO TO 25
      A(I, (J-1)) = MU2*(I-1)
      IF (I.EQ.N) GO TO 20
      IF (NUM.LE.0) GO TO 61
      A(I, (J+1)) = LAMDA2
      NUM=NUM-1
      C SPECIAL CASE AZERO SAVE
      61 IF (K.EQ.0) GO TO 60
      IF (I.EQ.1) GO TO 65
      A(I,J) = -1.0*((I-1)*MU2+LAMDA1+A(I, (J+1)))+AMAT(I)
      GO TO 20
      65 A(I,J) = -1.0*(LAMDA2+LAMDA1+AMAT(I))
      C
      GO TO 20
      60 IF (I.EQ.1) GO TO 70
      A(I,J) = -1.0*(LAMDA1+A(I, (J+1)))+(I-1)*MU2
      GO TO 20
      70 A(I,J) = -1.0*(LAMDA1+A(I, (J+1)))

```

```

20  CONTINUE
80  A(N,N)=LAMBDA1
    IF(K.NE.0)GO TO 95
    A((N-1),N)=0.0
95  RETURN
    END
    SUBROUTINE BLDMAT(KK,AMAT)
C*****
C THIS SUBROUTINE BUILDS M(N) INVERSE.
C*****
    IMPLICIT REAL*8(A-H,C-Z)
    REAL*4 LAMBDA,M1,M2,MCC,MU1,MU2,LAMBDA1,LAMBDA2
    COMMON /AREA1/ MU1,MU2,LAMBDA1,LAMBDA2,N,NSLOTS,MLIMIT
    DIMENSION AMAT(50)
    DO 10 I=1,N
15  K=N-I
    IF(KK.LT.K)GO TO 16
    IF(K.LE.0)GO TO 17
    AMAT(I)=K*MU1
    GO TO 10
17  AMAT(I)=MU1
    GO TO 10
10  AMAT(I)=K*MU1
    CONTINUE
    IF(KK.NE.MLIMIT)AMAT(N)=-1.0*(NSLOTS*MU2+LAMBDA1)
    IF(KK.EQ.MLIMIT)AMAT(N)=-1.0*(NSLOTS*MU2)
    RETURN
    END
    SUBROUTINE LINSYS(P,N,X)

```

```

C*****
C THIS ROUTINE SOLVES MATRIX P TO OBTAIN P(0) ELEMENTS
C*****
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION P(50,50),X(50)
      ITOP=2*N
      DO 10 K=1,N
        TMP=1.0/P(K,K)
      C DIVIDE ROW ELEMENTS BY DIAGONAL
      DO 20 J=K,N
        P(K,J)=P(K,J)*TMP
      CONTINUE
      X(K)=X(K)*TMP
      DO 30 J=1,N
        IF(K.EQ.J)GO TO 30
        TMP=P(J,K)
      C ZERO OUT ROW ELEMENTS ABOVE/BELOW DIAGONAL
      DO 40 L=K,N
        P(J,L)=P(J,L)-P(K,L)*TMP
      CONTINUE
      X(J)=X(J)-X(K)*TMP
      CONTINUE
      CONTINUE
      WRITE(5,200)
      DO 50 K=1,ITOP,8
        L=K+7
        WRITE(6,201) (X(I),I=K,L)
      CONTINUE
      RETURN
C

```

```
200  FORMAT(' ', 'PROBABILITY VECTOR P0 IS')  
201  FORMAT(' ', 'R(E14.6,1X)')  
      END
```

APPENDIX C

Appendix C contains analytic model output for various problem representations. System steady-state measures can be observed as NSLOTS and MLIMIT are allowed to vary. These measures are clearly identified in the output.

```

LAMBDA1=300.0000 LAMBDA2=
5.0000 MU1=600.0000 MU2=
6.3300 CHANNELS= 20 LIMIT= 30

```

PASS NUMBER	2	0.02500	1.00003	0.02632	1.00006	0.02778
PASS NUMBER	3	0.03125	1.00019	0.03333	1.00024	0.03571
PASS NUMBER	4	0.04165	1.00047	0.04543	1.00058	0.04997
PASS NUMBER	5	0.06245	1.00110	0.07135	1.00140	0.08322
PASS NUMBER	6	0.06245	1.00110	0.07135	1.00140	0.08322
PASS NUMBER	7	0.12469	1.00374	0.16605	1.00659	0.24836
PASS NUMBER	8	0.0	0.0	0.0	0.0	0.0
PASS NUMBER	9	0.0	0.0	0.0	0.0	0.0
PASS NUMBER	10					
PASS NUMBER	11					
PASS NUMBER	12					
PASS NUMBER	13					
PASS NUMBER	14					
PASS NUMBER	15					
PASS NUMBER	16					
PASS NUMBER	17					
PASS NUMBER	18					
PASS NUMBER	19					
EIGENVALUES ARE						
	1.00000					
	1.00014					
	1.00038					
	1.00088					
	1.00251					
	0.97847					

[illegible]

```

0.0
PROBABILITY MATRIX 30
0.26426D-06 0.50568D-05 0.28939D-04 0.20713D-03 0.56914D-03 0.21830D-02
0.20895D-01 0.34859D-01 0.52970D-01 0.72845D-01 0.91930D-01 0.10712D 00
0.11071D 00 0.98442D-01 0.81523D-01 0.57664D-01 0.19272D-01 0.0
0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0
0.58439D 00 0.29286D 00 0.76864D-01 0.15946D-01 0.39700D-02 0.16751D-02
0.73171D-03 0.67361D-03 0.63968D-03 0.61750D-03 0.60115D-03 0.58780D-03
0.55497D-03 0.54513D-03 0.53562D-03 0.52640D-03 0.51744D-03 0.50872D-03
0.48198D-03 0.47618D-03 0.46859D-03 0.46123D-03 0.44646D-03 0.10372D-01
EXPECTED DATA TRANSACTIONS IN SYSTEM 0.10301082D 01
EXPECTED DATA TRANSACTIONS IN QUEUE 0.53512807D 00
EXPECTED VOICE CALLS IN SYSTEM 0.14198302D 02
PROB OF ALL SLOTS USED BY VOICE 0.19272140D-01
DATA CONDITIONED ON FULL VOICE 0.21180817D 02
PROB VOICE USING MORE THAN C-2 SLOTS 0.15845870D 00
DATA CONDITIONED ON VOICE >C-2 0.38398148D 01

```

LAMDA1=300.0000 LAMDA2= 5.0000 MU1=600.0000 MU2= 0.3300 CHANNELS= 20 LIMIT= 50

PASS NUMBER	2
PASS NUMBER	3
PASS NUMBER	4
PASS NUMBER	5
PASS NUMBER	6
PASS NUMBER	7
PASS NUMBER	8
PASS NUMBER	9
PASS NUMBER	10
PASS NUMBER	11
PASS NUMBER	12
PASS NUMBER	13
PASS NUMBER	14
PASS NUMBER	15
PASS NUMBER	16
PASS NUMBER	17
PASS NUMBER	18
PASS NUMBER	19

Variable	Mean	Standard deviation	Skewness	Kurtosis	Normality test
Age	30.2500	0.02500	1.00003	0.02632	1.00006
Gender	1.00014	0.03125	1.00019	0.03333	1.00024
Marital status	1.00038	0.04165	1.00047	0.04543	1.00058
Education	1.00098	0.06245	1.00110	0.07135	1.00140
Income	1.00251	0.12469	1.00374	0.16605	1.00659
Health	0.97347	0.0	0.0	0.0	0.0

PROBABILITY VECTOR PO IS

[illegible]

PROBABILITY MATRIX	50	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.27827D-06	0.48770D-05	0.31133D-04	0.14525D-03	0.60437D-03	0.21186D-02				
0.20847D-01	0.34896D-01	0.52364D-01	0.72832D-01	0.91993D-01	0.10720D 00				
0.11045D 00	0.98577D-01	0.81612D-01	0.57445D-01	0.18727D-01	0.0				
0.0	0.0	0.0	0.0	0.0	0.0				
0.0	0.0	0.0	0.0	0.0	0.0				
0.0	0.0	0.0	0.0	0.0	0.0				
0.58414D 01	0.29247D 00	0.76635D-01	0.15802D-01	0.38750D-02	0.16048D-02				
0.68206D-03	0.62533D-03	0.59201D-03	0.57006D-03	0.55374D-03	0.54032D-03				
0.50696D-03	0.49688D-03	0.4712D-03	0.47762D-03	0.46835D-03	0.45932D-03				
0.43349D-03	0.42530D-03	0.41729D-03	0.40949D-03	0.40187D-03	0.39444D-03				
0.37323D-03	0.36650D-03	0.35995D-03	0.35356D-03	0.34734D-03	0.34128D-03				
0.32401D-03	0.31856D-03	0.31326D-03	0.30809D-03	0.30307D-03	0.29819D-03				
0.27958D-03	0.64951D-02	0.0	0.0	0.0	0.0				

EXPECTED DATA TRANSACTIONS IN SYSTEM C.12802253D 01
 EXPECTED DATA TRANSACTIONS IN QUEUE C.78327331D 00
 EXPECTED VOICE CALLS IN SYSTEM C.14194890D 02
 PROB OF ALL SLOTS USED BY VOICE C.18727385D-01
 DATA CONDITIONPD ON FULL VOICE C.29659480D 02
 PROB VOICE USING MORE THAN C-2 SLOTS C.15778447D 00
 DATA CONDITIONED ON VOICE >C-2 C.54326201D 01

LANDA1=300.0000 LANDA2= 5.0000 MU1=600.0000 MU2= 0.3300 CHANNPLS= 48 LIMIT= 60

PASS NUMBER	2
PASS NUMBER	3
PASS NUMBER	4
PASS NUMBER	5
PASS NUMBER	6
PASS NUMBER	7
PASS NUMBER	8
PASS NUMBER	9
PASS NUMBER	10
PASS NUMBER	11
PASS NUMBER	12
PASS NUMBER	13
PASS NUMBER	14
PASS NUMBER	15
PASS NUMBER	16
PASS NUMBER	17
PASS NUMBER	18
PASS NUMBER	19
PASS NUMBER	20
PASS NUMBER	21
PASS NUMBER	22
PASS NUMBER	23
PASS NUMBER	24
PASS NUMBER	25
PASS NUMBER	26
PASS NUMBER	27
PASS NUMBER	28
PASS NUMBER	29
PASS NUMBER	30
PASS NUMBER	31
PASS NUMBER	32

EXPECTED DATA TRANSACTIONS IN QUEUE 0.69634625D-04
EXPECTED VOICE CALLS IN SYSTEM 0.15152783D 02
PROB OF ALL SLOTS USED BY VOICE 0.0
DATA CONDITIONED ON FULL VOICE 0.0
PROB VOICE USING MORE THAN C-2 SLOTS 0.0
DATA CONDITIONED ON VOICE >C-2 0.0

LAMDA1=300.0000 LAMDA2= 5.0000 W1=600.0000 W2= 0.3300 CHANNELS= 48 LIMIT= 70

PASS NUMBER 2
PASS NUMBER 3
PASS NUMBER 4
PASS NUMBER 5
PASS NUMBER 6
PASS NUMBER 7
PASS NUMBER 8
PASS NUMBER 9
PASS NUMBER 10
PASS NUMBER 11
PASS NUMBER 12
PASS NUMBER 13
PASS NUMBER 14
PASS NUMBER 15
PASS NUMBER 16
PASS NUMBER 17
PASS NUMBER 18
PASS NUMBER 19
PASS NUMBER 20
PASS NUMBER 21
PASS NUMBER 22
PASS NUMBER 23
PASS NUMBER 24
PASS NUMBER 25
PASS NUMBER 26
PASS NUMBER 27
PASS NUMBER 28
PASS NUMBER 29
PASS NUMBER 30
PASS NUMBER 31
PASS NUMBER 32

EXPECTED DATA TRANSACTIONS IN SYSTEM 0.501145160 00
EXPECTED DATA TRANSACTIONS IN QUEUE 0.164902610-03
EXPECTED VOICE CALLS IN SYSTEM 0.151527500 02
PROB OF ALL SLOTS USED BY VOICE 0.0
DATA CONDITIONED ON FULL VOICE 0.0
PROB VOICE USING MORE THAN C-2 SLOTS 0.0
DATA CONDITIONED ON VOICP >C-2 0.0

APPENDIX D

Appendix D contains summary information from a simulation run of a 10-node network. Individual channel, packet, and circuit node statistics are detailed.

SYSTEM PARAMETERS

NODES LINKS SLOTS RATIO SLOT NODE CS PS MSJ START TIME END TIME PACKET VDR RATES
 TIME DELAY ARRIVAL APPRIAL
 10 MS 50 MS 10 MIN 10 SEC 0 MS 480090 MS 99000 320000000

SLOT	PACKETS SENT	CHANNEL 1 UTILIZATION		NUMBER OF VOICE CALLS
		PER CENT UTILIZATION	PER CENT UTILIZATION	
1	654	0.97	0.97	1
2	2776	0.91	0.91	1
3	3545	0.95	0.95	5
4	2911	0.87	0.87	1
5	2392	0.81	0.81	3
6	2440	0.94	0.94	1
7	3063	0.81	0.81	3
8	1199	0.85	0.85	1
9	1477	0.93	0.93	2
10	1094	0.85	0.85	1
11	921	0.92	0.92	3
12	2517	0.63	0.63	4
13	5102	0.55	0.55	1
14	6839	0.47	0.47	0
15	3753	0.64	0.64	2
16	3452	0.63	0.63	4
17	65	0.83	0.83	1
18	1259	0.72	0.72	0
19	2826	0.49	0.49	3
20	3005	0.39	0.39	0
21	1499	0.39	0.39	1
22	3292	0.30	0.30	0
23	1913	0.45	0.45	2
24	448	0.63	0.63	0

25	3989	0.18	0
26	930	0.45	1
27	1117	0.32	1
28	619	0.34	0
29	2069	0.11	1
30	1517	0.06	0
31	1523	0.05	0
32	443	0.11	0
33	0	0.19	0
34	454	0.16	0
35	533	0.09	0
36	705	0.02	0
37	430	0.02	0
38	124	0.01	0
39	55	0.01	0
40	96	0.00	0
41	86	0.00	0
42	85	0.00	0
43	0	0.0	0
44	0	0.0	0
45	0	0.0	0
46	0	0.0	0
47	0	0.0	0
48	0	0.0	0

TOTAL PACKETS SPNT 73113 TOTAL LINK UTILIZATION 0.39

SLOT	PACKETS SENT	CHANNEL 2 UTILIZATION		NUMBER OF VOICE CALLS
		PER CENT UTILIZATION		
1	1334	0.96		2
2	752	0.97		2
3	2394	0.92		4

4	1991	0.94	4
5	1453	0.96	2
6	1404	0.94	2
7	3949	0.89	3
8	3593	0.89	2
9	4875	0.82	3
10	2296	0.91	1
11	6297	0.75	2
12	2872	0.87	1
13	5307	0.76	1
14	2591	0.94	2
15	4107	0.75	0
16	4052	0.74	2
17	5038	0.65	3
18	2076	0.78	2
19	2132	0.76	1
20	5219	0.62	2
21	3179	0.72	3
22	2644	0.90	1
23	2692	0.68	1
24	3675	0.76	2
25	2174	0.69	1
26	2387	0.67	3
27	1321	0.71	1
28	2701	0.60	1
29	4300	0.60	2
30	7571	0.29	1
31	1512	0.64	2
32	3850	0.42	2
33	797	0.59	0
34	3380	0.28	2
35	1277	0.61	1
36	2020	0.43	1

37	2347	0.18	2
38	1226	0.45	1
39	1719	0.24	0
40	1209	0.34	0
41	1130	0.18	1
42	1748	0.05	0
43	1237	0.04	0
44	193	0.25	0
45	105	0.32	0
46	635	0.02	0
47	162	0.00	0
48	95	0.00	0

TOTAL PACKETS SENT 120975 TOTAL LINK UTILIZATION 0.59

SLOT	PACKETS SENT	CHANNEL 3 UTILIZATION PER CENT UTILIZATION	NUMBER OF VOICE CALLS
1	436	0.98	5
2	697	0.98	1
3	723	0.96	1
4	1850	0.88	3
5	2819	0.83	3
6	2103	0.90	2
7	5793	0.78	2
8	3354	0.82	1
9	808	0.93	1
10	2091	0.86	2
11	2216	0.96	2
12	4899	0.73	5
13	2252	0.84	2
14	3650	0.76	1
15	922	0.88	0

16	2252	0.80	2
17	3291	0.68	2
18	2450	0.72	2
19	3011	0.73	1
20	4010	0.63	3
21	2925	0.53	3
22	700	0.78	2
23	2824	0.66	1
24	1229	0.65	1
25	3331	0.57	1
26	3391	0.49	0
27	1043	0.58	1
28	4542	0.35	1
29	2956	0.37	1
30	443	0.54	0
31	2629	0.35	2
32	4011	0.13	0
33	2872	0.10	0
34	760	0.41	0
35	1727	0.10	1
36	1218	0.06	0
37	791	0.03	0
38	572	0.02	0
39	467	0.01	0
40	95	0.32	0
41	204	0.01	0
42	169	0.00	0
43	84	0.00	0
44	84	0.00	0
45	0	0.0	0
46	0	0.0	0
47	0	0.0	0
48	0	0.0	0

TOTAL PACKETS SENT		88860		TOTAL LINK UTILIZATION		0.49	
SLOT	PACKETS SENT	CHANNEL	UTILIZATION	PER CENT UTILIZATION	NUMBER OF VOICE CALLS		
1	513			0.98	2		
2	1219			0.97	1		
3	1748			0.94	2		
4	2239			0.93	3		
5	3425			0.89	3		
6	2756			0.90	1		
7	1236			0.95	2		
8	1947			0.94	3		
9	1095			0.93	3		
10	356			0.97	1		
11	1552			0.94	4		
12	1701			0.88	6		
13	826			0.92	1		
14	1416			0.88	2		
15	265			0.85	3		
16	1588			0.85	3		
17	1351			0.84	1		
18	4016			0.73	3		
19	3286			0.76	3		
20	1127			0.86	0		
21	6953			0.64	0		
22	1020			0.83	2		
23	622			0.89	2		
24	2060			0.79	0		
25	2104			0.77	1		
26	1823			0.73	0		
27	953			0.71	3		
28	1792			0.71	0		

29	414	0.82	1
30	1206	0.78	1
31	1790	0.73	2
32	511	0.76	0
33	2447	0.64	1
34	3380	0.63	3
35	924	0.62	1
36	2659	0.52	1
37	2444	0.72	2
38	1659	0.62	1
39	1400	0.60	0
40	2248	0.63	3
41	3058	0.48	0
42	3579	0.35	2
43	1716	0.45	2
44	1358	0.49	1
45	760	0.51	1
46	1022	0.53	1
47	1507	0.36	0
48	148	0.47	0

TOTAL PACKETS SENT 95230 TOTAL LINK UTILIZATION 0.74

SLOT	PACKETS SENT	CHANNEL 5 UTILIZATION		NUMBER OF VOICE CALLS
		PER CENT UTILIZATION		
1	316	0.98		2
2	1180	0.94		3
3	395	0.96		1
4	917	0.93		3
5	978	0.95		2
6	1611	0.89		1
7	2784	0.84		2

3	2208	0.87	1
9	2963	0.82	2
10	1541	0.89	4
11	761	0.91	1
12	1515	0.85	1
13	1599	0.84	1
14	443	0.90	1
15	2790	0.73	4
16	838	0.83	2
17	1576	0.79	3
18	425	0.84	0
19	1552	0.67	1
20	3300	0.66	0
21	1712	0.72	2
22	2292	0.66	1
23	4163	0.62	4
24	3195	0.56	1
25	344	0.66	0
26	1132	0.73	2
27	784	0.60	0
28	5695	0.29	1
29	3227	0.38	1
30	1063	0.58	0
31	1900	0.47	0
32	1749	0.73	1
33	1239	0.45	1
34	1073	0.45	1
35	2922	0.36	1
36	2984	0.26	1
37	2518	0.17	1
38	227	0.47	0
39	1139	0.28	0
40	1673	0.06	0

41	51	0.41	0
42	1100	0.04	0
43	413	0.28	0
44	765	0.03	0
45	465	0.02	0
46	410	0.01	0
47	86	0.01	0
48	86	0.01	0

TOTAL PACKETS SENT 74499 TOTAL LINK UTILIZATION 0.57

SLOT	PACKETS SENT	CHANNEL 6 UTILIZATION		NUMBER OF VOICE CALLS
		PER CENT UTILIZATION	PER CENT UTILIZATION	
1	685	0.97	0.97	3
2	643	0.94	0.94	1
3	1235	0.99	0.99	2
4	2769	0.78	0.78	4
5	2723	0.93	0.93	2
6	3363	0.71	0.71	1
7	2893	0.66	0.66	1
8	3567	0.72	0.72	2
9	1214	0.73	0.73	2
10	276	0.87	0.87	0
11	3151	0.58	0.58	2
12	722	0.83	0.83	2
13	2283	0.53	0.53	1
14	5433	0.53	0.53	1
15	1616	0.58	0.58	1
16	2320	0.51	0.51	1
17	3524	0.42	0.42	2
18	1045	0.52	0.52	0
19	3923	0.28	0.28	1

20	1996	0.35	0
21	258	0.53	0
22	3432	0.14	0
23	1152	0.31	0
24	3258	0.11	0
25	923	0.28	0
26	2155	0.07	0
27	2156	0.08	0
28	1503	0.05	0
29	1398	0.04	0
30	1187	0.04	0
31	657	0.02	0
32	455	0.01	0
33	455	0.01	0
34	0	0.38	0
35	96	0.00	0
36	97	0.00	0
37	87	0.00	0
38	11	0.00	0
39	11	0.00	0
40	11	0.00	0
41	12	0.00	0
42	12	0.00	0
43	12	0.00	0
44	0	0.0	0
45	0	0.0	0
46	0	0.0	0
47	0	0.0	0
48	0	0.0	0
TOTAL PACKETS SENT		646.11	TOTAL LINK UTILIZATION
		0.32	

SLOT	PACKETS SENT	CHANNEL 7 PER CENT UTILIZATION	UTILIZATION PER CENT UTILIZATION	NUMBER OF VOICE CALLS
1	1412	0.96	0.96	2
2	1560	0.94	0.94	3
3	1415	0.94	0.94	2
4	1293	0.95	0.95	2
5	1751	0.93	0.93	3
6	1062	0.96	0.96	3
7	2329	0.94	0.94	3
8	2782	0.88	0.88	2
9	4351	0.85	0.85	3
10	6113	0.76	0.76	3
11	1704	0.88	0.88	1
12	3324	0.91	0.91	2
13	4162	0.81	0.81	3
14	6742	0.79	0.79	1
15	1663	0.79	0.79	3
16	5123	0.76	0.76	1
17	4070	0.78	0.78	2
18	2451	0.85	0.85	1
19	3562	0.77	0.77	1
20	3711	0.74	0.74	3
21	3442	0.73	0.73	2
22	2516	0.78	0.78	2
23	5128	0.63	0.63	1
24	3955	0.66	0.66	2
25	2663	0.73	0.73	2
26	6233	0.46	0.46	2
27	1655	0.69	0.69	2
28	1041	0.78	0.78	2
29	1610	0.66	0.66	2
30	1298	0.62	0.62	2
31	4546	0.44	0.44	0

32	3997	0.42	1
33	1594	0.57	1
34	1305	0.49	1
35	2236	0.39	0
36	3268	0.48	2
37	1700	0.43	0
38	1591	0.37	0
39	2377	0.32	1
40	4415	0.14	0
41	3579	0.14	0
42	943	0.29	0
43	1614	0.17	1
44	1747	0.15	0
45	665	0.25	1
46	300	0.25	0
47	1752	0.04	0
48	1123	0.03	0

TOTAL PACKETS SENT 130344 TOTAL LINK UTILIZATION 0.61

SLOT	PACKETS SENT	CHANNEL 8 UTILIZATION PER CENT UTILIZATION	UTILIZATION NUMBER OF VOICE CALLS
1	517	0.97	3
2	916	0.94	2
3	1679	0.93	2
4	3163	0.84	5
5	2733	0.86	2
6	2937	0.83	2
7	2412	0.86	2
8	2462	0.90	3
9	1701	0.89	1
10	2570	0.83	1

11	5065	0.56	1
12	2356	0.79	1
13	5731	0.60	1
14	2199	0.81	1
15	6211	0.56	2
16	458	0.84	1
17	2292	0.71	4
18	4323	0.40	3
19	1610	0.59	2
20	3734	0.38	0
21	3644	0.36	0
22	1489	0.47	0
23	1654	0.49	3
24	4707	0.15	0
25	2257	0.30	1
26	364	0.49	0
27	1033	0.23	1
28	837	0.31	1
29	574	0.32	0
30	1627	0.24	0
31	831	0.24	0
32	1231	0.22	0
33	1413	0.20	0
34	1364	0.19	0
35	1722	0.05	0
36	108	0.22	0
37	1312	0.06	0
38	1270	0.04	0
39	848	0.02	0
40	180	0.01	0
41	135	0.00	0
42	10	0.00	0
43	0	0.0	0

44	0	0.0	0
45	0	0.0	0
46	0	0.0	0
47	0	0.0	0
48	0	0.0	0

TOTAL PACKETS SENT 43940 TOTAL LINK UTILIZATION 0.41

SLOT	PACKETS SENT	CHANNEL 9 UTILIZATION		NUMBER OF VOICE CALLS
		PER CENT UTILIZATION		
1	1074	0.94		5
2	474	0.98		3
3	314	0.96		5
4	2867	0.90		5
5	1717	0.94		1
6	2289	0.93		2
7	1308	0.94		1
8	1650	0.91		2
9	1486	0.93		3
10	1064	0.94		2
11	2046	0.89		2
12	1206	0.92		2
13	4743	0.76		3
14	3582	0.84		2
15	2097	0.89		1
16	3596	0.83		3
17	1227	0.90		2
18	5490	0.75		2
19	5343	0.71		2
20	4575	0.70		1
21	807	0.84		2
22	748	0.85		1

23	4090	0.66	2
24	3143	0.70	1
25	303	0.87	1
26	2023	0.74	2
27	2790	0.71	2
28	3008	0.64	0
29	1413	0.65	1
30	3554	0.44	3
31	1343	0.64	1
32	1464	0.77	2
33	3748	0.60	2
34	1498	0.51	1
35	2740	0.50	0
36	363	0.98	1
37	934	0.60	3
38	2591	0.49	1
39	2511	0.43	1
40	1423	0.50	1
41	278	0.50	0
42	464	0.50	0
43	1680	0.25	1
44	545	0.60	3
45	1272	0.31	2
46	130	0.58	0
47	663	0.45	0
48	965	0.19	0

TOTAL PACKETS SENT 95529 TOTAL LINK UTILIZATION 0.71

SLOT	PACKETS SENT	CHANNEL 10 UTILIZATION PER CENT UTILIZATION	NUMBER OF VOICE CALLS
1	1544	0.94	3

2 1 3 4 2 2 5 2 0 1 3 2 2 3 1 2 1 3 3 1 1 1 1 4 0 1 2 1 2 2 2 1 2

0.98
0.97
0.93
0.92
0.91
0.93
0.83
0.94
0.97
0.75
0.67
0.81
0.91
0.87
0.90
0.92
0.77
0.78
0.73
0.80
0.67
0.78
0.80
0.80
0.86
0.61
0.66
0.59
0.66
0.60
0.77
0.67
0.53

805
1063
1987
2386
2125
1963
2222
756
116
6302
8105
2675
505
1128
552
223
3153
2716
3725
1563
4129
2425
1772
594
2927
3405
3076
2590
2064
296
2863
3849

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34

35	3403	0.43	1
36	3296	0.37	1
37	1073	0.48	1
38	549	0.68	2
39	763	0.50	0
40	2417	0.31	1
41	1450	0.36	0
42	3315	0.16	0
43	1214	0.28	2
44	2433	0.15	0
45	1552	0.16	1
46	1396	0.06	0
47	386	0.13	0
48	367	0.01	0

TOTAL PACKETS SENT 99321 TOTAL LINK UTILIZATION 0.65

SLOT	PACKETS SENT	CHANNEL 11 UTILIZATION PER CENT	UTILIZATION PER CENT	NUMBER OF VOICE CALLS
1	670	0.97	0.97	2
2	1783	0.91	0.91	3
3	1558	0.94	0.94	3
4	209	0.97	0.97	0
5	1882	0.90	0.90	3
6	3132	0.79	0.79	4
7	2486	0.82	0.82	2
8	3032	0.77	0.77	2
9	632	0.89	0.89	1
10	747	0.30	0.30	2
11	2827	0.66	0.66	4
12	3424	0.74	0.74	2
13	5013	0.64	0.64	2

14	1934	0.79	2
15	2697	0.77	3
16	2941	0.68	1
17	5479	0.51	1
18	1469	0.74	2
19	0	0.86	0
20	3700	0.56	1
21	911	0.79	1
22	2333	0.66	1
23	1241	0.67	0
24	764	0.65	1
25	2093	0.46	0
26	4809	0.24	1
27	2607	0.29	1
28	1486	0.46	1
29	1463	0.58	1
30	2465	0.22	1
31	2125	0.18	0
32	753	0.21	0
33	140	0.34	0
34	833	0.15	0
35	1121	0.08	0
36	1187	0.04	0
37	750	0.03	0
38	241	0.09	0
39	191	0.08	0
40	291	0.01	0
41	218	0.01	0
42	28	0.00	0
43	28	0.00	0
44	28	0.00	0
45	0	0.0	0
46	0	0.0	0

47 0 0.0 0
48 0 0.0 0

TOTAL PACKETS SENT 73696 TOTAL LINK UTILIZATION 0.46

SLOT	PACKETS SENT	CHANNEL 12 UTILIZATION PER CENT UTILIZATION	UTILIZATION NUMBER OF VOICE CALLS
1	621	0.94	2
2	1573	0.94	3
3	3990	0.87	4
4	2193	0.91	2
5	4395	0.78	4
6	2905	0.87	2
7	742	0.96	1
8	3031	0.83	0
9	2544	0.85	2
10	1248	0.91	1
11	1751	0.87	2
12	4122	0.74	1
13	2651	0.80	2
14	1971	0.90	2
15	971	0.83	1
16	4905	0.59	0
17	777	0.85	2
18	3999	0.67	4
19	2833	0.70	2
20	5300	0.49	2
21	3335	0.65	2
22	763	0.62	1
23	2419	0.62	2
24	3583	0.51	3
25	386	0.86	1

25	2975	0.47	1
27	1319	0.48	0
28	2426	0.28	0
29	1158	0.69	1
30	304	0.55	2
31	1798	0.22	1
32	54	0.58	0
33	11	0.64	0
34	13	0.58	0
35	1572	0.24	1
36	1739	0.14	1
37	1660	0.05	0
38	732	0.24	1
39	825	0.03	0
40	911	0.02	0
41	711	0.02	0
42	633	0.01	0
43	213	0.00	0
44	78	0.00	0
45	0	0.0	0
46	0	0.0	0
47	0	0.0	0
48	0	0.0	0

TOTAL PACKETS SENT 82041 TOTAL LINK UTILIZATION 0.52

PACKET NODE STATISTICS

NODE	DELAY (SECS)	<.1	<.2	<.3	<.4	<.5	<.6	<.7	<.8	<.9
1		35464	11395	182	0	0	140	0	38	283
2		37370	19562	83	0	0	0	0	0	0
3		39887	9999	309	204	196	0	122	110	199
4		29478	19910	407	0	0	0	156	162	0
5		26099	21533	457	0	0	126	0	127	0

NODE	CURRENT PACKET LOAD	CUMULATIVE PACKET LOAD	CUMULATIVE TRANSACTIONS
1	568	48491	5628
2	156	50057	5742
3	575	51877	6097
4	279	50841	5859
5	359	49801	5862

NODE	DST	CURRENT PACKET LOAD	CURRENT TRANSACTION LOAD	CUMULATIVE PACKET LOAD	LOAD
1	2	312	25	12131	12131
1	3	54	9	11831	11831
1	4	202	20	12102	12102
1	5	0	0	12427	12427
2	1	124	16	11928	11928
2	3	32	8	13620	13620
2	4	0	0	13025	13025
2	5	0	0	11484	11484
3	1	161	13	12778	12778
3	2	172	18	15157	15157
3	4	109	17	13046	13046
3	5	133	11	10896	10896
4	1	51	6	12470	12470
4	2	41	7	14564	14564

13079
10728
14033
13165
10374
12229

19
9
25
10
10
4

138
49
230
55
49
25

3 5 1 2 3 4

4 4 5 5 5 5

CIRCUIT SWITCH NODE STATISTICS

NODE	TOTAL CALLS	CALLS LOST	BLOCKING	CALLS IN SYSTEM
6	72	0	0.0	30
7	39	0	0.0	25
8	86	0	0.0	38
9	51	0	0.0	27
10	74	0	0.0	14

APPENDIX E

The data contained in Appendix E supports the graphical presentations shown in Chapter VII. Each table contains summary information of network behavior for a specified arrival pattern. Headings and titles make each table self explanatory.

Table E1. Arrival Pattern 1

$$\lambda_1 = 50/\text{sec}$$

$$\lambda_2 = 5/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	38874	.18
2	56391	.29
3	43905	.23
4	55470	.39
5	39195	.29
6	21405	.13
7	55293	.29
8	38283	.17
9	55429	.37
10	55308	.33
11	34686	.23
12	40902	.27
Avg	44595	.26

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	.10	1
2	.10	1
3	.10	1
4	.10	2
5	.10	1
Avg	.10	1.2

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	43	0.0	14
7	52	0.0	18
8	44	0.0	12
9	27	0.0	10
10	41	0.0	10
Avg	41.4	0.0	12.8

Table E2. Arrival Pattern 2

$$\lambda_1 = 200/\text{sec}$$

$$\lambda_2 = 5/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	150564	.26
2	227247	.41
3	173043	.32
4	215493	.51
5	141996	.36
6	79431	.17
7	230826	.41
8	161001	.26
9	215424	.50
10	216744	.45
11	138147	.30
12	152466	.35
Avg	175198	.35

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	.10	5
2	.10	3
3	.10	3
4	.10	4
5	.10	2
Avg	.10	3.4

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	43	0.0	14
7	52	0.0	18
8	44	0.0	12
9	27	0.0	10
10	41	0.0	10
Avg	41.4	0.0	12.8

Table E3. Arrival Pattern 3

$$\lambda_1 = 350/\text{sec}$$

$$\lambda_2 = 5/\text{min}$$

Link Usage		
Link	Throughput	Utilization
1	256002	.35
2	393978	.54
3	299781	.42
4	363798	.63
5	251445	.44
6	149385	.23
7	409920	.53
8	283980	.35
9	374978	.56
10	367775	.62
11	242091	.38
12	276579	.43
Avg	305809	.45

Packet Node Summary		
Node	Packet Delay (Sec)	Data in System
1	.11	8
2	.10	7
3	.10	6
4	.11	5
5	.10	4
Avg	.104	6.0

Circuit Node Summary			
Node	Total Calls	Blocking	Calls in System
6	43	0.0	14
7	52	0.0	18
8	44	0.0	12
9	27	0.0	10
10	41	0.0	10
Avg	41.4	0.0	12.8

Table E4. Arrival Pattern 4

$$\lambda_1 = 500/\text{sec}$$

$$\lambda_2 = 5 \text{ min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	384150	.44
2	574194	.67
3	418002	.49
4	492114	.71
5	370248	.52
6	243537	.31
7	590823	.67
8	410283	.44
9	503589	.71
10	532458	.68
11	390666	.49
12	397665	.52
Avg	442310	.55

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	.15	16.5
2	.14	5.2
3	.18	12.9
4	.21	13.1
5	.21	11.2
Avg	.18	11.8

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	43	0.00	14
7	52	0.00	18
8	44	0.02	11
9	27	0.00	10
10	41	0.02	10
Avg	41.4	0.008	12.6

Table E5. Arrival Pattern 5

$$\lambda_1 = 100/\text{sec}$$

$$\lambda_2 = 10/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	73113	.39
2	120975	.59
3	88866	.49
4	85230	.74
5	74499	.57
6	64611	.32
7	130344	.61
8	83940	.41
9	95529	.71
10	99321	.65
11	73686	.46
12	82041	.52
Avg	89346	.53

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	.18	6
2	.14	3
3	.13	6
4	.19	5
5	.23	5
Avg	.17	5.0

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	72	0.0	30
7	89	0.0	25
8	86	0.0	38
9	61	0.0	27
10	74	0.0	14
Avg	76.4	0.0	26.8

Table E6. Arrival Pattern 6

$$\lambda_1 = 200/\text{sec}$$

$$\lambda_2 = 10/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	163716	.45
2	251625	.68
3	175707	.55
4	165300	.78
5	141675	.61
6	129624	.39
7	251550	.69
8	172224	.46
9	175611	.76
10	203487	.72
11	175125	.54
12	164157	.57
Avg	180816	.60

Packet Node Summary

<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	.22	14.1
2	.15	8.7
3	.29	16.0
4	.29	11.7
5	.35	15.2
Avg	.26	13.1

Circuit Node Summary

<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	72	0.03	27
7	89	0.00	27
8	86	0.02	37
9	61	0.02	26
10	74	0.00	12
Avg	76.4	0.01	25.8

Table E7. Arrival Pattern 7

$$\lambda_1 = 300/\text{sec}$$

$$\lambda_2 = 10/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	245532	.59
2	333993	.78
3	258927	.61
4	271506	.79
5	226833	.67
6	179883	.49
7	358662	.77
8	286050	.57
9	302925	.77
10	330669	.77
11	252708	.66
12	252672	.66
Avg	275030	.68

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	.45	25.4
2	.26	17.3
3	.60	26.3
4	.46	25.5
5	.61	27.0
Avg	.47	24.3

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	72	0.00	24
7	89	0.00	28
8	86	0.08	29
9	61	0.05	25
10	74	0.08	16
Avg	76.4	0.04	24.4

Table E8. Arrival Pattern 8

$$\lambda_1 = 500/\text{sec}$$

$$\lambda_2 = 10/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	521970	.73
2	598008	.87
3	445971	.73
4	402108	.87
5	380637	.78
6	374433	.63
7	544392	.87
8	475761	.73
9	398043	.88
10	513417	.85
11	544041	.76
12	473937	.78
Avg	472726	.79

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	1.82	35.0
2	1.48	32.4
3	2.01	34.2
4	2.35	40.5
5	2.37	28.7
Avg	2.00	34.1

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	72	0.07	27
7	89	0.06	19
8	86	0.09	28
9	61	0.08	22
10	74	0.16	11
Avg	76.4	0.09	21.4

Table E9. Arrival Pattern 9

$$\lambda_1 = 50/\text{sec}$$

$$\lambda_2 = 15/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	49572	.59
2	62040	.77
3	35181	.70
4	36879	.84
5	29583	.75
6	31167	.61
7	50163	.79
8	35790	.61
9	38484	.82
10	46110	.81
11	51849	.69
12	50662	.70
Avg	43123	.72

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	1.85	51.6
2	2.21	85.6
3	2.69	79.2
4	2.67	72.0
5	2.62	54.2
Avg	2.40	68.5

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	108	0.10	31
7	137	0.19	51
8	124	0.19	39
9	104	0.20	37
10	118	0.19	31
Avg	118.2	0.17	37.8

Table E10. Arrival Pattern 10

$$\lambda_1 = 200/\text{sec}$$

$$\lambda_2 = 15/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	149400	.72
2	218595	.85
3	135270	.76
4	73443	.88
5	121179	.80
6	158946	.68
7	207996	.85
8	156879	.73
9	124815	.86
10	142020	.86
11	195174	.78
12	174612	.78
Avg	154860	.79

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	2.89	39.7
2	2.73	89.7
3	3.20	36.2
4	2.80	37.5
5	2.87	86.3
Avg	2.89	57.8

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	108	0.14	27
7	137	0.20	51
8	124	0.19	36
9	104	0.27	30
10	118	0.21	32
Avg	118.2	0.20	35.2

Table E11. Arrival Pattern 11

$$\lambda_1 = 350/\text{sec}$$

$$\lambda_2 = 15/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	309231	.79
2	358560	.88
3	256956	.81
4	214221	.90
5	171807	.83
6	229458	.74
7	325479	.88
8	291339	.80
9	285081	.89
10	324252	.88
11	337392	.81
12	324237	.82
Avg	285667	.83

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	3.23	30.9
2	3.16	15.1
3	3.60	29.8
4	3.92	36.8
5	3.31	40.4
Avg	3.44	30.6

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	108	0.24	26
7	137	0.18	40
8	124	0.29	28
9	104	0.28	30
10	119	0.24	32
Avg	118.4	0.246	31.2

Table E12. Arrival Pattern 12

$$\lambda_1 = 100/\text{sec}$$

$$\lambda_2 = 20/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	55695	.78
2	71718	.87
3	52125	.82
4	24966	.90
5	43584	.85
6	55491	.78
7	55320	.88
8	43218	.79
9	47319	.91
10	33864	.89
11	41406	.84
12	61134	.85
Avg	48820	.846

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	4.44	92.4
2	3.50	221.4
3	4.01	221.4
4	2.53	231.1
5	3.40	227.6
Avg	3.57	198.7

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	150	0.29	42
7	188	0.44	49
8	173	0.47	35
9	151	0.43	42
10	162	0.47	22
Avg	164.8	0.42	38

Table E13. Arrival Pattern 13

$$\lambda_1 = 200/\text{sec}$$

$$\lambda_2 = 20/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	137514	.82
2	200823	.89
3	155250	.83
4	65091	.91
5	63375	.87
6	146799	.79
7	216063	.89
8	158361	.82
9	131850	.91
10	167364	.88
11	171363	.84
12	118095	.87
Avg	144329	.86

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	4.39	49.6
2	4.22	133.3
3	4.93	40.6
4	4.71	52.9
5	4.34	117.1
Avg	4.51	78.7

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	150	0.38	44
7	188	0.40	44
8	173	0.44	41
9	151	0.50	24
10	162	0.46	24
Avg	164.8	0.43	35.4

Table E14. Arrival Pattern 14

$$\lambda_1 = 300/\text{sec}$$

$$\lambda_2 = 20/\text{min}$$

Link Usage		
<u>Link</u>	<u>Throughput</u>	<u>Utilization</u>
1	207393	.85
2	303951	.91
3	297009	.85
4	143895	.92
5	86013	.88
6	181269	.81
7	354591	.90
8	291639	.84
9	266826	.92
10	316137	.91
11	192012	.87
12	131673	.89
Avg	231034	.88

Packet Node Summary		
<u>Node</u>	<u>Packet Delay (Sec)</u>	<u>Data in System</u>
1	4.58	22.3
2	4.76	61.2
3	4.68	33.1
4	4.58	19.8
5	4.95	46.8
Avg	4.71	36.64

Circuit Node Summary			
<u>Node</u>	<u>Total Calls</u>	<u>Blocking</u>	<u>Calls in System</u>
6	150	0.45	37
7	188	0.43	42
8	173	0.45	31
9	151	0.60	24
10	162	0.48	30
Avg	164.8	0.482	32.8

VITA

Carroll A. Clabaugh was born in Red Oak, Iowa, on November 30, 1940, to Clinton and Arlene Clabaugh. He graduated from Coe College with a B.A. in Mathematics in 1963. In June 1963, he was commissioned a Second Lieutenant in the United States Air Force. Later, in 1971, he graduated from New Mexico State University with a M.S. in Computing Science. The author has been on continuous active duty since 1963, and is currently on the promotion list for Lieutenant Colonel. He has completed the Squadron Officers' School, Air Command and Staff School, Air War College and Communications' Staff Officer School. His forthcoming assignment is as a Staff Communications' Officer within the Operations Directorate of the Defense Communications Agency, located in Arlington, Virginia. Awards and decorations include the Air Force Commendation and Meritorious Service medals. Major Clabaugh is a member of ACM, IEEE, and the UPE Honor Society. His permanent mailing address is 522 Greene Street, Boone, Iowa 50036.

END

DATE
FILMED

12-81

DTIC

